# Optimization Methods for Modeling Diversity in Language Technologies

by

Sachin Kumar

CMU-LTI-23-013

**Thesis Committee**
Yulia Tsvetkov (chair), University of Washington
Emma Strubell, Carnegie Mellon University
Graham Neubig, Carnegie Mellon University
Shuly Wintner, University of Haifa
Chris Dyer, Google DeepMind

Submitted in partial conformity with the requirements
for the degree of Doctor of Philosophy

Graduate Department of Language Technologies Institute
Carnegie Mellon University

# Abstract

Language use varies across individuals, communities, and populations giving rise to different variations with diverging vocabularies, syntax, semantics, and pragmatics. Despite rapid improvements in natural language processing systems on standard benchmarks in several languages, these models often fail to represent this diversity. In this thesis, I aim to develop methods to make NLP systems understand and generate natural languages, while explicitly modeling extra-linguistic variables associated with diverse language use.

Reformulating conventional training and inference problems in neural network-based NLP models as instances of multi-objective optimization, this thesis is divided into two parts. In the first part, (a) I present a method to train robust text classification models demoting reliance on spurious correlations in data – with applications to detecting language varieties as well as other tasks where patterns of variation are confounds; (b) I present a prompting framework to contextualize text classifiers for pragmatic tasks to different domains, and social and personal factors of variation. In the second part, I focus on enriching diversity in text generation. I present (c) a training algorithm for machine translation that separates token representation learning from model learning resulting in improved lexical diversity in the generated text. We show that it lends to easy adaptability to generate closely related dialects of the target language. Finally, I present (d) decoding algorithms to control for stylistic variations from pretrained language models. I frame controlled decoding as constrained optimization and develop gradient-based methods to generate text non-autoregressively which initialize and update the entire output sequence iteratively. We validate these approaches with different types of controls on machine translation, style transfer, and open-ended generation. Overall, this thesis aims to advance research directions in NLP beyond standardized language towards societal use, where research questions and methodology are guided by relevant training and inference objectives.

To my parents

# Contents

# Chapter 1

# Introduction

No two speakers of a particular language exhibit identical speech patterns. Ever-growing user-generated web data has brought such variability to the surface. This abundance of raw text, along with recent advances in large-scale self-supervised learning methods (Devlin et al., 2019; Radford et al., 2019a; Brown et al., 2020a; Chowdhery et al., 2022) have resulted in remarkable performance improvements on many NLP tasks. However, despite the diversity of training data, NLP models tend to be monolithic, encoding only the frequent patterns and averaging over all other variations (Hovy and Prabhumoye, 2021). As a result, they are static and brittle—consistently failing to support language use outside the traditional "standard". For example, toxicity detection systems score tweets in African American English (AAE) as more offensive than others (Davidson et al., 2019; Sap et al., 2019), sentiment classification systems rate reviews by women authors as more positive (Kiritchenko and Mohammad, 2018), and text generation models have higher error rates for women and dialectal speakers (Tatman, 2017; Ziems et al., 2023), have inconsistent personalities in conducting dialogues with humans (Cercas Curry et al., 2020), and generate culturally inappropriate and impolite outputs (Vanmassenhove et al., 2019; Hovy et al., 2020).

Prior NLP work related to language variability has largely focused on developing learning approaches to essentially *ignore* the differences among variations, treating them as noise. Examples of such works include improving part-of-speech taggers (Gimpel et al., 2011), dependency parsers (Liu et al., 2018), sentiment analyzers (Yang and Eisenstein, 2017), named entity recognizers (Augenstein et al., 2017) or translation systems (Michel and Neubig, 2018) where the input text is from, say social media containing different varieties. However, in several cases, the differences are intentional, and their intended meaning can be deduced by other language users; ignoring these signals can hurt model utility. With the increasing adoption of NLP systems in user-facing products, there is consequently an urgent need to *focus on language variability*. For example, building dialogue agents that adapt to users' linguistic preferences as opposed to producing responses in standardized languages for everyone, translation systems that can generate outputs in diverse styles and fluency levels; building classification models that do not ignore sources of variations but take them into account to make predictions.

Towards this goal, this thesis focuses on machine learning solutions that rethink standard training and inference methods for language understanding and generation. The unifying theme is consolidation of linguistic and extra-linguistic diversity in text at varying granularities into ML models: from lexicon (Kumar and Tsvetkov, 2019; Bhat et al., 2019; Kumar et al., 2021a), to syntax and style (Jegadeesan et al., 2021;

Kumar et al., 2021b, 2022b; Han et al., 2023a), to semantics and pragmatics (Kumar et al., 2023b).[1] I argue that, unlike conventional algorithms for training and inference, which optimize model parameters and outputs towards a single objective (such as maximizing task accuracy, or generating a faithful translation of an input sentence), *characterizing variation in NLP is inherently multi-objective*. For example, maximizing task accuracy *and* fairness across speaker varieties; generating a faithful translation *and* controlling for a specific style, *and* controlling for simplicity (or complexity) of linguistic structure. In addition, to promote usage in broad practical settings, this thesis aims to develop efficient solutions; from computational complexity, to the number of parameters, to amount of labeled resources.

## 1.1 Thesis Statement

My central goal in this thesis is to develop language technologies that reflect and support the diversity of their users, by building models which are fair towards diverse populations and adaptive to their linguistic attributes and preferences. My primary methodology to achieve this involves questioning standard training and inference algorithms in such models and reinterpreting them as instances of multi-objective optimization where I seek to consolidate linguistic and extra-linguistic diversity in text at varying granularity. In the first part of the thesis, I focus on text classification problems where source of variation may interfere or inform the task. I present: (a) a method to train robust text classification models demoting reliance on spurious correlations in data – with applications to detecting language varieties as well as other tasks where patterns of variation are confounds; and (b) a prompting framework to contextualize text classifiers for pragmatic tasks to different domains, and social and personal factors of variation. In the second part, I focus on enriching diversity in text generation. I present (c) a training algorithm for machine translation that separates token representation learning from model learning resulting in improved lexical diversity in the generated text. We show that it lends to easy adaptability to generate closely related dialects of the target language. Finally, I present (d) decoding algorithms to control for stylistic variations from pretrained language models. I frame controlled decoding as constrained optimization and develop gradient-based methods to generate text non-autoregressively which initialize and update the entire output sequence iteratively. I demonstrate the capabilities of our proposed solutions on a plethora of NLP applications including language identification (Kumar et al., 2019a), sentiment analysis (Kumar et al., 2023b), hate speech classification (Xia et al., 2020), machine translation (Kumar and Tsvetkov, 2019; Bhat et al., 2019; Kumar et al., 2021a), prompted story generation (Kumar et al., 2022b), summarization, and paraphrasing (Kumar et al., 2021b). These solutions are general and grounded in theory. They advance machine learning research, and have much wider applicability, across data domains and tasks.

## 1.2 Thesis Overview

I start with the relevant background (Chapter 2) on standard training and inference methods in NLP, optimization, and a brief overview of language variation that puts the research in this thesis in context. Subsequently, this thesis is divided into two parts based on application areas: text classification and text generation. Each part is further organized based on the kinds of variations, their respective challenges and accompanying training or inference methods to address them.

[1] Variation at the phonological level is a challenging and important problem to study in its own right. However, the focus of this thesis is only on written variations in language.

## Part I: Language variation and text classification

With the goal of maximizing task accuracy, the basic assumption in text classification is that a single training objective can evaluate the overall performance of the model. However, for many NLP tasks modeled as text classification, content and task labels may both correlate with signals of variation in the input text. In Chapter 3, to achieve better generalization across different variations in text domains and styles, we amend this goal to not only maximize task accuracy but also minimize dependence on spurious correlations. Representing this goal as an adversarial objective, we developed a two-phase alternating optimization algorithm that learns to disentangle content from language variety markers and demotes such spurious confounds in the learned representations. With experiments across several tasks and datasets, this approach results in improvement in out-of-domain generalization for detecting language varieties.

**Relevant Paper**: Kumar et al. (2019a)

Studies in language pragmatics have argued that linguistic knowledge, such as knowledge of grammar and vocabulary, alone is usually insufficient for social interactions.Interlocutors often rely on background assumptions and draw on extralinguistic knowledge of the world. While humans learn this knowledge by being in social situations, it is not encoded in raw text. Towards understanding these issues, in Chapter 4, we develop an approach towards personalizing classification models which do not predict the label in isolation but rely on extralinguistic information. Building on the popular paradigm of language model prompting, we propose a multivariate generative prompting framework. Our method measures the LM likelihood of input text conditioned on natural language descriptions of labels, enabling seamless integration of various additional contextual information to improve task performance. On various standard classification benchmarks, with three open-source LM families, we show that zero-shot classification with simple contextualization of the data source of the evaluation set consistently outperforms both zero-shot and few-shot baselines while improving robustness to prompt variations.

**Relevant Paper**: Kumar et al. (2023b)

## Part II: Language variation and text generation

In the second part of this thesis, towards enriching linguistic diversity in language generation, I propose methods to address, (a) lexical variation, and (b) stylistic or syntactic variation.

Most language generation models generate outputs using a fixed vocabulary. Commonly used frequency-based tokenization schemes to construct this vocabulary come at a cost of ignoring lexical diversity, in addition to making the models less amenable to adaptation to new varieties with different distribution of words.

In Chapter 5, to overcome these limitations, I propose to separate lexical representation learning and model learning with distinct objectives. We introduce a method for training text generation models, by predicting pre-trained word vectors instead of softmax probabilities with a new training loss based on hyperspherical distributions, enabling an open and dynamic vocabulary. We learned these vectors with an auxiliary objective (using internal structure of words)that leads to richer representations. With experiments on machine translation (MT), we showed improved accuracy for rare words, improving lexical diversity in the generated text, especially in morphologically rich languages. Furthermore, we showed that this approach lends to rapid adaptation of MT models to generate close dialects of the target language with little monolingual dialectal data.

**Relevant Papers**: Kumar and Tsvetkov (2019), Bhat et al. (2019), Jegadeesan et al. (2021), Kumar et al. (2021a)

Finally, several factors influence how we write and are in fact predictable from written text.However,

language models trained on raw text without labels for such variations do not inherently allow to control for them. In Chapter 6, I introduce a method of controllable decoding from pretrained language models, which does not require re-training or fine-tuning them. We formulate decoding as a constrained optimization problem with maximizing the language model likelihood as the primary objective with each desirable attribute in the output set as constraints, which can be learned with a small amount of labeled data. With a soft-relaxation of this discrete optimization over the vocabulary and gradient descent to solve the optimization, we enable multi-attribute fine-grained control in many conditional generation tasks including machine translation and paraphrasingwith attributes such as author's demographic attributes (e.g. age), formality, and others. We further improve this approach to enable sampling from language models under constraints to enable further diversity in the generated text by by borrowing key ideas from Bayesian learning which rely on gradient based sampling methods. Additionally, we perform these gradient steps in the token embedding space to improve decoding speed and memory requirements.

**Relevant Papers**: Kumar et al. (2021b), Kumar et al. (2022a)

Following the main content chapters, Chapter 7 discusses the ethical implications considered in this work, and Chapter 8 concludes by highlighting the themes common across application domains as areas for future work.

# Chapter 2

# Background

This chapter contextualizes the research presented in this thesis. First, I present a brief overview of standard training and inference methods for text classification and generation. Next, I provide a brief overview of different kinds of variations within a language. Finally, I discuss multi-objective optimization different flavors of which form the basis of the solutions presented in each chapter.

## 2.1 Standard Training and Inference Methods in NLP

### 2.1.1 Text classification

Text classification, the task of assigning a label or class to a given text or collection of texts, is the most studied problem in NLP. A myriad of NLP tasks and applications can be formulated as text classification such as email spam detection, sentiment analysis, hate speech detection, natural language inference, among others.

Given an input text $\mathbf{x}$, the output (or target) $y$ is typically chosen from a predefined set $\mathcal{Y} = \{0, \dots C-1\}$, where $C$ is the number of classes, e.g. $0 = \texttt{not hate speech}$ and $1 = \texttt{hate speech}$. Neural network based classifiers model this task probabilistically as $P(y|\mathbf{x};\theta)$, the conditional distribution of the target $y$ given the input text. $\theta$ denotes the parameters of the neural network model. This set up also referred to as **discriminative classification** as the goal of the model is to discriminate between different labels. Traditionally, classification problems are formulated as a form of supervised learning, where we are given data as pairs of input text and output labels, $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots (\mathbf{x}_N, y_N)\}$. The parameters $\theta$ of the model $P$ can then estimated by maximizing the likelihood of $\mathcal{D}$ under $P$. In other words, for every training input $\mathbf{x}_i$, the model outputs a multinomial distribution over the set of labels $\mathcal{Y}$ and the goal is to minimize the following objective, also known as cross-entropy loss.

$$\text{CE} = \sum_{i=1}^{N} \sum_{c=1}^{C} -\log P(y_c|\mathbf{x}_i)\mathbb{I}(y_c = y_i)$$

Since this objective does not admit a closed-form solution with neural network based architectures, the parameters of the model are updated via stochastic gradient descent.

In recent years, with the advent of general-purpose language models pretrained on just raw text (Devlin et al., 2019; Radford et al., 2019a; Brown et al., 2020a; Chowdhery et al., 2022), text classification models of today range from fine-tuned versions of such pretrained models to the language models directly used in few-shot or zero-shot manner, where given a handful of demonstrations as additional input, the models can

reasonably predict the label for new test examples. The probabilistic classification model in such cases can modified as $P_{\text{LM}}(y|\mathbf{x}, y_1, \mathbf{x}_1, \ldots, y_K, \mathbf{x}_K)$, where $K \geq 0$ are the number of demonstration. $P_{\text{LM}}$ denotes language model probabilities. We discuss how the language (generation) models are trained next.

### 2.1.2 Text generation

Text generation is the task of synthesizing text that appears indistinguishable to human-written text to fulfill a communicative goal. It is also often referred to "natural language generation" in the literature. Several tasks and applications fall under this umbrella such as machine translation, paraphrasing, text simplification, dialogue generation, text summarization, story generation to name a few.

Typically, text generation models, are either realized as (1) encoder-decoder models where a distinct encoder neural networks takes as input some context that will aid the task, such as a source sentence for translation, a user utterance for dialogue generation, and the decoder generates the output text, or (2) decoder-only models where any additional context is simply input to the decoder itself. Both kinds of models can be expressed probabilistically as $P(\mathbf{y}|\mathbf{x}; \theta)$ where $\mathbf{y} = y_1, \ldots, y_N$ is an output sequence and $\mathbf{x}$ is the input (which may or may not be text) where each $y_i \in \mathcal{V}$, a predefined output vocabulary. Traditionally, given an input $\mathbf{x}$, decoding from such a model involves finding output(s) $\mathbf{y} \in \mathcal{Y}$ which admit a high probability under $P$. In practice, naively searching $\mathcal{Y}$ to find the highest probability generation is intractable as the space of possible sequences grows exponentially with sequence length. Hence, traditionally $P$ is factorized over each token $y_n$, where the output is generated left-to-right one token at a time. Such models are referred to as **autoregressive** language models, where given a generated or given context $\mathbf{c}_n$ at time step $n$, the model outputs a multinomial distribution over the output vocabulary using which a token $w_n$ is generated. It is then appended to the previous context to create a longer context $\mathbf{c}_{n+1} = [c_n, w_n]$ for the next token to be generated at time step $n + 1$. Several search and sampling strategies such as beam search, top-k sampling (Fan et al., 2018), and nucleus sampling (Holtzman et al., 2020), among others (Meister et al., 2023; Wiher et al., 2022) have been proposed in the literature to massage the output probabilities improve the quality of generated outputs.

Given a corpus (either supervised or raw), text generation models are again trained to maximize the likelihood of the corpus under the model distribution $P$ which is factorized per token as

$$P(\mathbf{y}|\mathbf{x}) = p(y_N|y_{1:N-1}, \mathbf{x})p(y_{N-1}|y_{1:N-2}, \mathbf{x}) \ldots p(y_1|\mathbf{x}).$$

That is, the models produce $N$ distributions over the output vocabulary for every sequence of length $N$ in the training corpus. The model parameters are trained to maximize the likelihood of these distributions for the sequence under consideration treating the generation of each token as a classification problem over the output vocabulary using a cross-entropy loss at every step. Again the parameters of the model are updated via stochastic gradient descent.

## 2.2 Linguistic Variations

Language as a medium of communication is inherently a social phenomenon. Since no two humans are alike, neither is their language use. Most of this variation is highly systematic and is manifested in pronunciation or accents of the speakers, their lexical choices, grammar or syntax of their utterances or even semantics or pragmatics depending on a number of non-linguistic factors. In this thesis, we only focus on written variations.

Following prior research (Labov, 1972; Fischer, 1958; Chambers, 1995), we broadly categorize them as follows:

**Interspeaker Variation** Variation between different speakers of a language, this dimension refers to (1) variation on a geographical level; a clear example of which is the existence of different **dialects** of the same language spoken in different regions, (2) variation determined by demographic attributes and social experiences of the speakers like age, gender, ethnicity, sexuality, socio-economic status, native language among others (Fischer, 1958; Eckert and McConnell-Ginet, 2003; Ferreira, 2007; Levon, 2007; Holmes and Wilson, 2017; Meyerhoff and Ehrlich, 2019). They are also referred to as **sociolects**. The speech of an old, non-educated woman, for example, will be different from the speech of a young, educated boy. And finally, (3) variations unique to individuals referred to as **idiolects**. The term has been abstractly defined as the totality of the possible utterances an individuals could say and have been studied in stylometry analysis and forensic linguistics (Wright, 2013; Coulthard, 2004)

**Intraspeaker Variation** Variation within the speech of a single speaker, this dimension is determined by social situations like the medium of communication (e.g., a phone call versus a lecture versus an email), attributes of or relationship with the interlocutor or the audience (for example, listeners' fluency, age, gender, intimacy level, and power dynamics). Examples of such variations include different levels of politeness and formality, complexity or simplicity of sentence structures, different levels of code-switching in multilingual communities, and so on. These variations are often referred to as styles in the NLP literature (Jin et al., 2022).

## 2.3 Optimization

Most machine learning algorithms involve minimizing an objective function.

$$\min_{\theta} J(\theta)$$

where $\theta$ are model weights. For training a supervised model, for instance, the objective function $J$ measures the discrepancy between model outputs and the intended target; the most common example of which is cross-entropy loss for classification as discussed previously. Since most modern machine learning models are defined using neural networks, closed form solutions of $J$ are impossible to obtain. And instead solutions are obtained via (stochastic) gradient descent on $\theta$.

One of the most important aspect to building such solutions is defining the objective function. However, most real-world problems require optimizing for multiple objectives,

$$\min_{w}(J_1(w), J_2(w), \ldots)$$

For example, in classification, $J_1$ can be cross-entropy as before and $J_2$ could be a regularization function to prevent overfitting. In many settings, different objectives may be at odds at one another and decreasing one may increase the other. That is, in multi-objective optimization, there may not typically exist a feasible solution that minimizes all objective functions simultaneously. What is desired are Pareto optimal solutions (Debreu, 1954); that is, solutions that cannot be improved in any of the objectives without degrading at least one of the other objectives.

Since we deal with neural networks based approaches, of particular relevance to this thesis is gradient-based multi-objective optimization (Qu et al., 2021). In each chapter presented next, we show that incorporating variations in NLP models involve optimizing for multiple objectives. However, since gradient updates can only be performed with one loss function, we apply different ways to train models and perform inference from them from devising a new training schedules (Chapter 3, Chapter 5) to optimize objectives to combining different objectives into one via linear combination or constrained optimization (Chapter 4, Chapter 6).

# Part I

# Language Variation and Text Classification

# Chapter 3

# Confound Invariant Text Classification

*This chapter discusses work previously published in Kumar et al. (2019b).*

Neural network based classification systems have been shown to be biased towards learning frequent spurious correlations in the training data that may be confounds in the task (Leino et al., 2019). For example, in sentiment classification, the term Spielberg may be correlated with the positive class because many of director Steven Spielberg's movies have positive reviews. However, the term itself does not indicate a positive review. A major challenge in building such systems is to discover features that are not just correlated with the signals in the training data, but are true indicators of these signals, and therefore generalize well.

In NLP systems, demographic attributes reflected in written text are often a source of such correlations, both in terms of language describing different demographics, as well as language generated by different populations. For example, Kiritchenko and Mohammad (2018); Shen et al. (2018) showed sentiment analysis systems implicitly overfit to gender of the author systematically amplifying the intensity ratings of posts written by women. Sap et al. (2019) showed that toxicity classifiers are more likely to predict posts containing African American Vernacular English (AAVE) as toxic. Conversely, classifiers with a goal to detect language varieties are confounded by the content of the input text rather than focusing on linguistic or extra-linguistic markers of variation. For example, we show in this chapter that a simple classifier trained to predict the native language (L1) of the author given English text written by them, is likely to predict that a person's L1 is Greek if the texts authored by that person mentions Greece. Field and Tsvetkov (2020) show that a classifier trained to detect intra-speaker variations corresponding to gender bias through predicting the gender of the addressee ends up relying on content rather than style of the author.

In this chapter, we present a general framework to train text classification models that demotes such spurious correlations. We address this problem in two steps. First, we introduce a method for representing *latent* confounds. Recent relevant work in the area of domain adaptation (Ganin et al., 2016) and deconfounding for text classification (Pryzant et al., 2018; Elazar and Goldberg, 2018) assumes that the set of confounds is known a priori, and their values are given as part of the training data. This is not always possible and limiting the applicability of such models. In contrast, we propose, based on log-odds ratio with Dirichlet prior (Monroe et al., 2008), a method for identifying and representing latent confounds as probability distributions. Second, we propose an alternating learning procedure with multiple adversarial discriminators, inspired by adversarial learning (Goodfellow et al., 2020), that demotes latent confounds and results in textual representations that are invariant to the confounds.

This framework is task-independent and can be extended to a vast array of text classification tasks where

confounding factors are not known a priori. In the published work presented in this chapter, we evaluate our approach on the task of *native language identification* (L1ID), which aims at automatically identifying the native language (L1) of an individual based on their language production in a second language (L2, English in this work). We experiment with two different datasets: a small corpus of student written essays (Malmasi et al., 2017) and a large and noisy dataset of Reddit posts (Rabinovich et al., 2018). The aim of this task is to discover stylistic features present in the input that are indicative of the author's L1. However, a model trained to predict L1 is likely to predict that a person is, say, a native Greek speaker, if the texts authored by that person mention Greece, because the training data exhibits such topical correlations. We show that classifiers trained on these datasets without any intervention learn these spurious topical correlations, and that our proposed deconfounded classifiers alleviate this problem. Follow-up work has also shown the utility of this method in identifying gender bias in text and demoting racial bias in text classification systems. We briefly discuss them towards the end.

## 3.1 Deconfounded Text Classification

We are given $N$ labeled documents in the training set $\{(\mathbf{x}_1, y_1, z_1), \ldots, (\mathbf{x}_N, y_N, z_N)\}$, where each $\mathbf{x}_i$ is a text document with the target label $y_i \in \mathcal{Y}$, and a spurious feature $z_i$. The spurious feature may be given a priori or may be latent. Throughout this chapter, we assume that $z_i$ is defined as a multinomial distribution of over $k$ variables. For example, for predicting hate speech, if the race of the author is a spurious feature, $z_i$ is represented as a one-hot vector representing the race of the author among $k$ race categories considered. Or, for predicting L1 (native language), if the content is spurious, $z_i$ can be a multinomial distribution over $k$ topics (say, learned from LDA; §3.2).

Given this data, we seek to train a classifier that predicts $\hat{y} = f_\theta(\mathbf{x})$ without relying on $z$. We input $\mathbf{x}$ to an encoder neural network $h(\mathbf{x}; \theta_h)$ to obtain a hidden representation $\mathbf{h_x}$ followed by two feedforward networks: (1) $c(h(\mathbf{x}); \theta_c)$ to predict the label $y$; and (2) an adversary network $\text{adv}(h(\mathbf{x}); \theta_a)$ to predict the spurious variable $z$. If $\mathbf{h_x}$ does not encode any information to predict $z$, then $c(h(\mathbf{x}))$ will not depend on $z$. Concretely, we want to optimize the following:

$$\min_{\theta_c, \theta_h} \frac{1}{N} \sum_{i=1}^{N} \text{CE}(c(\mathbf{h}_{\mathbf{x}_i}), y_i)$$
$$\text{s.t., } \text{adv}_{\theta_a}(\mathbf{h}_{\mathbf{x}_i}) = \mathbb{U}_K \forall i, \forall \theta_\text{a}$$

where CE denotes cross-entropy loss and $\mathbb{U}_K = (\frac{1}{K}, \ldots, \frac{1}{K})$. That is, any classifier $\mathbf{adv}$ trying to predict the spurious features $z$ should be confused (and predict a uniform distribution over all possible values of the feature). To make this constrained optimization problem feasible to solve via gradient descent, we modify it to two optimization problems shown below that we alternately optimize for (with $\lambda$ as a hyperparameter).

$$\min_{\theta_c, \theta_h} \frac{1}{N} \sum_{i=1}^{N} \lambda \text{CE}(c(\mathbf{h}_{\mathbf{x}_i}), y_i) + (1 - \lambda) \text{CE}(\text{adv}_h^*(\mathbf{h}_{\mathbf{x}_i}), \mathbb{U}_K) \tag{3.1}$$

$$\text{adv}_h^* = \arg\min_{\theta_a} \frac{1}{N} \sum_{i=1}^{N} \text{CE}(\text{adv}(\mathbf{h}_{\mathbf{x}_i}), z_i), \tag{3.2}$$

**Learning Schedule: Alternating Optimization of Classifier and Adversary** The model is trained by minimizing these two objectives in an alternating fashion (we use $\lambda = 0.5$ in all our experiments). The training

schedule is critical in adversarial setups where the loss has two competing terms (Mescheder et al., 2018; Arjovsky and Bottou, 2017; Roth et al., 2017); here, these terms minimize classification loss while maximizing the topic prediction loss. Algorithm 1 details our proposed alternating learning procedure.

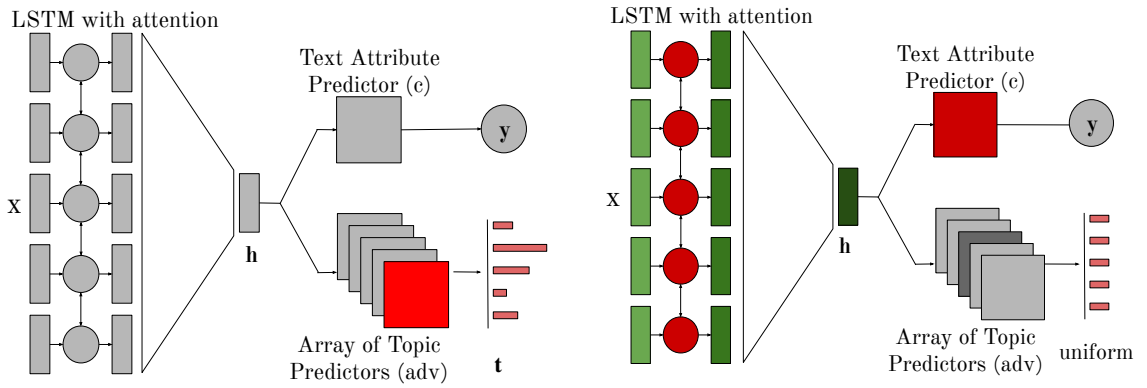
**Algorithm 1:** Alternating optimization of classifier and adversary.

**Result:** $\theta_h, \theta_c, \theta_{a_1}, \ldots, \theta_{a_T}$
Randomly initialize $\theta_h, \theta_c$;
**while** *not converged* **do**
  Sample a minibatch of $b$ training samples;
  Update $\theta_h$ and $\theta_c$ using gradients with
  respect to $\frac{1}{b}\sum_{i=1}^{b} \text{CE}(c(h(x_i)), y_i)$.;
$j = 1$;
**for** *number of training iterations* $T$ **do**
  Randomly initialize $\theta_{a_j}$;
  **for** $t$ *steps* **do**
    Sample a minibatch of $b$ training samples;
    Fix $\theta_h$ and $\theta_c$, update $\theta_{a_j}$ using gradients with respect to $\frac{1}{b}\sum_{i=1}^{b} \text{CE}(\text{adv}_{\theta_{a_j}}(h(x_i)), t_i)$;
  **for** $c$ *steps* **do**
    Sample a minibatch of $b$ training samples;
    Fix $\theta_{a_u}$ for $u \in_R \{1, \ldots, j\}$ and update $\theta_c$ and $\theta_h$ using gradients with respect to
    $\frac{1}{b}\sum_{i=1}^{b} \text{CE}(c(h(x_i)), y_i) + \text{CE}(\text{adv}_{\theta_{a_u}}(h(x_i)), \mathbb{U}_K)$;
  $j \longleftarrow j + 1$;

inspired by generative adversarial networks (GANs; Goodfellow et al., 2020) (see Figure 3.1). First (*pretraining*), we train the encoder along with the classifier using only the classification loss, until convergence. After pretraining, $\mathbf{h_x}$ encodes spurious information which it uses for classification. Now, we train only $\text{adv}(\mathbf{h_x})$ to (accurately) predict $z$, keeping the parameters of $h(.)$ fixed. Once $\text{adv}(.)$ is trained, it should be able to predict the spurious feature from $\mathbf{h}_x$ with high accuracy (spurious feature training). The goal now is to modify $\mathbf{h}_x$ in such a way that $\text{adv}(\mathbf{h}_x)$ produces a uniform distribution (that is, fooling the adversary; similar to fooling the discriminator in GANs). We do that by keeping the weights of $\text{adv}(.)$ fixed, and training the network to produce the class label (via $c(\cdot)$) and a uniform distribution (via $\text{adv}(\cdot)$; *spurious feature forgetting*). We then repeat this procedure for a fixed number of steps which was tuned using the validation set.

**Multiple Adversaries** In our experiments, we observe that after every "forgetting" stage, $\text{adv}(.)$ does end up producing a uniform distribution, but in the next "training" phase, $\text{adv}(.)$ is able to reproduce the spurious feature accurately. We hypothesize that, during forgetting, the classifier learns to encode the spurious feature in a different way than it did in the previous step. Inspired by the "experience replay" approach used in reinforcement learning (O'Neill et al., 2010; Mnih et al., 2015), we propose using *multiple adversaries*. During the $i$th "spurious feature training" phase, we train a new adversary $\text{adv}_i$ (with parameters $\theta_{a_i}$ instead of retraining only one adversary over and over again. In the next "forgetting" phase, at each training step we pick $\text{adv}_j$ at random from the pool of previously learned adversaries, $j \in_R \{1, \ldots, i\}$. By using multiple adversaries, we make it difficult for the classifier to encode spurious information anywhere.

First, we evaluate this approach on the task of native language identification or L1ID (§3.2). Second, we briefly discuss extensions of this approach to detect gender bias in text and demote racial bias in hate-speech detection §3.6.




(a) Weights of the LSTM and of the discriminator are fixed. A new topic predictor is trained by minimizing the cross entropy of the output and the distribution of the input document over latent topics.

(b) Weights of all the topic predictors are fixed, but the encoder is trained. The model is jointly minimizing the cross-entropy of the classifier and encouraging the topic predictor toward uniformity.

Figure 3.1: We alternate between training the topic predictor (left) and the deconfounded classifier/encoder (right). Pretraining is not shown in the figure.

## 3.2 Native Language Identification

To detect variations defined by native language of the speaker, we study L1ID, in which the goal is to predict the native language (L1) of a writer given text authored by them in English (their L2).

### 3.2.1 Motivational Case Study

We study the general effect of *topical* confounds in text classification. To motivate the need to demote them, we introduce as a case study the L1ID task, in which the goal is to predict the native language of a writer given their texts in L2.

We begin with a subset of the L2-Reddit corpus (Rabinovich et al., 2018), consistsing of Reddit posts by authors with 23 different L1s, most of them European languages. Some of the posts come from Europe-related forums (e.g. r/Europe, r/AskEurope, r/EuropeanCulture), whereas others are from unrelated forums. We view the latter as out-of-domain data and use them to evaluate the generalization of our models. We use a subset of this corpus, with only the 10 most frequent L1s, to guarantee a large enough balanced training set. We remove all the posts with fewer than 50 words and sample the dataset to obtain a balanced distribution of labels: from this balanced dataset, we randomly sample 20% of examples from each class and divide them equally to create development and test sets. In total, there are around 260,000 examples in the training set and 32,000 examples each in the development, the in-domain test set, and the out-of-domain test set.

We trained a standard (non-adversarial) classifier, with a bidirectional LSTM encoder followed by two feedforward layers with a $\tanh$ activation function and a softmax in the final layer (full experimental details are given in §3.3.2). We refer to this model as NO-ADV. The results are shown in Table 3.1. Notice the huge drop in accuracy on the out-of-domain data, which indicates that the model is learning topical features.

To further verify this claim, we used *log-odds ratio with Dirichlet prior* (Monroe et al., 2008)—a common way to identify words that are statistically overrepresented in a particular population compared to others—to identify the top-$K$ words that were most strongly associated with a specific L1 in the training set. (We refer the reader to (Monroe et al., 2008) for the details about the algorithm.) We experimented with $K \in$

$\{20, 50, 100, 200\}$. Table 3.2 shows the top-10 words in each class; observe that almost all of these words are geographical (hence, topical) terms that have nothing to do with the L1.

Next, we masked such topical words (by replacing them with a special token) and evaluate the trained classifier on masked test sets. Accuracy (Table 3.1) degrades on both the in-domain and out-of-domain sets, even when only 20 words are removed. The drop in accuracy with the out-of-domain dataset is smaller since these data do not include many instances where the presence of topical words would help in identifying the label. These experiments confirm our hypothesis that the baseline classifier is primarily learning topical correlations, and motivate the need for a deconfounded classification approach which we describe next.

| | **In-Domain** | **Out-of-Domain** |
|---|---|---|
| **NO-ADV** | 52.5 | 25.7 |
| **+MASK TOP-20** | 32.8 | 21.0 |
| **+MASK TOP-50** | 31.6 | 20.4 |
| **+MASK TOP-100** | 30.1 | 19.7 |
| **+MASK TOP-200** | 28.5 | 18.7 |

Table 3.1: Motivation: accuracy (%) of L1ID on the L2-Reddit dataset.

| | |
|---|---|
| **English** | ireland irish british britain russia scotland england states american london brexit |
| **Finnish** | finland finnish finns helsinki swedish finn nordic sweden sauna nokia estonian |
| **French** | french france paris sarkozy macron fillon hollande gaulle hamon marine valls breton |
| **German** | german germany austria merkel refugees asylum germans bavaria austrian berlin also |
| **Greece** | greek greece greeks syriza macedonia athens turkey macedonians fyrom turkish ancient |
| **Dutch** | dutch netherlands amsterdam wilders rotterdam holland rutte belgium bike hague |
| **Polish** | poland polish poles warsaw lithuanian lithuania judges jews ukranians imho tusk |
| **Romanian** | romania romanian romanians moldova bucharest hungarian hungarians transistria |
| **Spanish** | spain catalan spanish catalonia catalans madrid barcelona independence spaniards |
| **Swedish** | sweden swedish swedes stockholm swede malmo danish nordic denmark finland |

Table 3.2: Top words based on log-odds scores for each label in the L2-Reddit dataset.

### 3.2.2 Representing Topical Confounds

For this task, preliminary analysis reveals that a vanilla classifier relies on content rather than writing style apparent by poor out-of-domain performance. However, unlike common applications of adversarial classifiers (such as domain adaptation), the confounding variable is not available during training. A common solution to learn topic distributions in a collection of documents is Latent Dirichlet allocation (LDA; Blei et al., 2003)–a probabilistic generative model for discovering abstract topics that occur in a collection of documents. Under LDA, each document can be considered a mixture of a small (fixed) number of topics—each represented as a distribution over words—and each word's presence is assumed to be attributed to one of the document's topics. More precisely, LDA assigns each document a probability distribution over a fixed number of topics $K$. However, LDA topics are known to be poor features for classification (McAuliffe and Blei, 2008), indicating that they do not encode all the topical information. Moreover, they can encode information that is not actually topical and can be a useful L1 marker. We observe this in our experiments as well.

Motivated by our case study, we propose representing these content or topical confounds using a "weak classifier" which is likely to utilize spurious correlations. We build this weak classifier using log-odds scores

(Monroe et al., 2008). For each class label $y$ and each word type $w$, we calculate a log-odds score $lo(w, y) \in \mathbb{R}$. The higher this score, the stronger the association between the class and the word. As we saw in §3.2.1, the highest scored words are mostly topical and hence constitute superficial features which we want the classification model to "unlearn." We therefore define a distribution which assigns high probability to a document containing these high scoring words. For a label $y \in \mathcal{Y}$ and an input document $x = \langle w_1, \ldots, w_n \rangle$, we define $p(y \mid x)$:

$$p(y \mid x) \propto p(y) \cdot p(x \mid y) = p(y) \cdot \prod_{i=1}^{n} p(w_i \mid y)$$

The above expansion assumes a bag of words representation. When the dataset is balanced, $p(y)$ is equal for each label and can be omitted. Finally, we define $p(w_i \mid y) \propto \sigma(lo(w_i, y))$, where $\sigma(.)$ is the sigmoid function, which squashes the log-odds scores (whose values are in $\mathbb{R}$) to the range $[0, 1]$. We normalize the sigmoid values over the vocabulary to convert them to a probability distribution. In this distribution, the number of "topics" equals the number of labels, $m$.

## 3.3 Experimental Setup

### 3.3.1 Datasets

We evaluate our topical confound demotion method on the L1ID task. We show experiments with two datasets where L2 is English: the L2-Reddit dataset described in §3.2.1, and TOEFL17, a collection of essays authored by non-native English speakers who apply for academic studies in the US (Malmasi et al., 2017). This corpus reflects eleven L1s: Arabic, Chinese, French, German, Hindi, Italian, Japanese, Korean, Spanish, Telugu, and Turkish. The training data include 11,000 authors (1,000 per L1) and the development set has 1,100 essays per L1. We evaluate on the development set. Each essay is also marked with a prompt ID which was given to the authors to write the essay. There are 8 prompts in total, based on which we construct 8 versions of train and test set. In each version, we remove essays marked with one of the prompts from both the train and the development sets, and consider the removed essays from the development set an "out-of-domain" test set. We refer to the version where prompt "P$K$" is out-of-domain as "–P$K$" in the results (Table 3.3), $K \in \{0, \ldots, 7\}$.

### 3.3.2 Implementation Details

We tokenized and lowercased all the text using spaCy. Limiting our vocabulary to the most frequent 30,000 words in the training data, we replaced all out-of-vocabulary words with "UNK." We encoded each word using a word embedding layer (initialized at random and learned) and passed these embeddings to a bidirectional LSTM encoder (one layer for each direction) with attention ($h(x)$; Pryzant et al., 2018). Each LSTM layer had a hidden dimension of 128. We used two layered feed forward networks with a $\tanh$ activation function in the middle layer (of size 256), followed by a softmax in the final layer, as $c(.)$ and $\mathrm{adv}(.)$.

### 3.3.3 Baselines

We consider several baselines that are intended to capture the stylistic features of the texts, explicitly avoiding content.

**Linear classifier with content-independent features (LR)** Replicating Goldin et al. (2018), we trained a logistic regression classifier with three types of features: function words, POS trigrams, and sentence length, all of which are reflective of the style of writing. We deliberately avoided using content features (e.g., word frequencies).

**Classification with no adversary on masked texts (LO-TOP-$K$)** We mask the top-$K$ words (based on log-odds scores) in *both* the train and the test sets (as in §3.2.1); we train the classification model again without training $\mathrm{adv}(.)$. After masking the top words, we expect patterns of writing style (and, therefore, L1) to become more apparent.

**Adversarial training with gradient reversal (GR-LO)** A common method of learning a confound-invariant representations is to use a gradient reversal layer (Beutel et al., 2017; Ganin et al., 2016; Pryzant et al., 2018; Elazar and Goldberg, 2018). The output of the encoder, $\mathbf{h}_x$, is passed through this layer before applying $\mathrm{adv}(.)$. This training setup usually proves too difficult to optimize, and often results in poor performance. That is, even if the performance of $\mathrm{adv}(.)$ is weak, $\mathbf{h}_x$ still ends up leaking information about the confound Lample et al. (2019a); Elazar and Goldberg (2018). In the forward pass, this layer acts as identity whereas in the backward pass it multiplies the gradient values by $-\lambda$, essentially reversing the gradients before they go into the encoder. $\lambda$ controls the intensity of the reversal (we used $\lambda = 0.2$).

**LDA topics as confounds (ALT-LDA)** We trained LDA on the training set and for each example in the training set, generated a probability distribution (over 50 topics), and used it as topical confound with our proposed learning setup, alternating classifier-adversary training.

## 3.4 Results

### 3.4.1 TOEFL17 Dataset

We begin with experiments on the TOEFL17 dataset, where predicting L1 is an easier task due to the lower proficiency of the authors. Table 3.3 reports the accuracy of our proposed model, denoted **ALT-LO**, compared to the logistic regression baseline (**LR**), and two adversarial baselines: one demotes latent log-odds-based topics via gradient reversal (**GR-LO**), and another uses our proposed novel learning procedure but demotes baseline LDA topics (**ALT-LDA**). We report both in-domain accuracy and out-of-domain results; the latter is obtained by averaging the accuracy of each set "–P$K$" over $K \in \{0, \ldots, 7\}$.

| | In-Domain | Out-of-Domain |
|---|---|---|
| **LR** | 55.3 | 50.9 |
| **GR-LO** | 12.7 | 13.6 |
| **ALT-LDA** | 59.1 | 50.1 |
| **ALT-LO** | **61.9** | **60.4** |

Table 3.3: Classification accuracy with topic-demoting methods, TOEFL dataset.

Our model strongly outperforms all baselines that demote confounds, in both classification setups. We observe in our experiments that gradient reversal is especially unstable and hyperparameter sensitive: it has been shown to work well with categorical confounds like domain type or binary gender, but in demoting

continuous outputs like a topic distribution, we observe it is not effective. The proposed alternating training with multiple discriminators obtains better results, and replacing LDA with log-odds-based topics also improves both in-domain and (much more substantially) out-of-domain predictions, confirming the effectiveness of our proposed innovations.

A vanilla classifier without demoting confounds (denoted in §3.2.1 as NO-ADV) yields in-domain and out-of-domain accuracies of 62.0 and 58.3, respectively. We would expect that the better generalization power of our proposed model would come at a price of lower accuracy in-domain. Our goal is to capture the true signals of L1, rather than superficial patterns that are more frequent in the data and artificially boost the performance in NO-ADV settings. This is indeed what we observe.

For example, the text "...i agree with you on the prolonged war if the plc heartland (poland proper) was not as rich as it was i dont really see how we would been ..." in the dataset is labeled as "Polish" instead of the gold label "Swedish" by the NO-ADV classifier, likely because of the mention of the term "poland", but the ADV-LO model predicts it correctly since it likely picks on other features that indicate non-fluency, like "we would been". Such naive classification errors become especially costly in making predictions about people's demographic attributes: ethnicity, which often correlates with L1, but also gender, race, religion, and others Hardt et al. (2016); Beutel et al. (2017).

### 3.4.2 L2-Reddit Dataset

Next, we experiment with L2-Reddit, a larger and more challenging dataset (since many speakers in the dataset are highly fluent, and the signal of their native language is weaker). The performance of the simple baselines on this dataset is shown in Table 3.4. The accuracy of the linear classifier is poor (compared to Table 3.1), perhaps because it fails to capture some contextual features learned by the neural network models. With LO-TOP-20, the performance on both test sets improves. It slightly degrades when more words are removed, perhaps because some words indicative of L1 are also removed.

| | **In-Domain** | **Out-of-Domain** |
|---|---|---|
| **LR** | 21.2 | 18.5 |
| **LO-TOP-20** | 38.7 | 21.9 |
| **LO-TOP-50** | 36.4 | 21.4 |
| **LO-TOP-100** | 35.8 | 21.2 |
| **LO-TOP-200** | 34.7 | 20.8 |

Table 3.4: Baseline classification accuracy on L2-Reddit.

Finally, we evaluate the impact of our novel training procedure and the quality of our proposed topical confound identification method. We compare our proposed solution, denoted ALT-LO, with two alternatives, as before, one with a different learning setup (GR-LO) and one with a different confound representation (ALT-LDA). Table 3.5 summarizes the results: our proposed learning procedure ALT-LO performs better than both the alternatives. Unsurprisingly, the model trained with gradient reversal (GR-LO) performs particularly poorly; this was our primary motivation to explore better learning techniques.

To further confirm that the ALT-LO model is not learning topical features, we repeat the experiment presented in Table 3.1—masking the top $K$ topical words (based on log-odds scores) from the test sets, but not retraining the models—now, with our proposed model ALT-LO. Table 3.6 shows that in contrast to standard models that do not demote topical confounds (as in Table 3.1), there is less degradation in the performance of

| | In-Domain | Out-of-Domain |
|---|---|---|
| GR-LO | 22.5 | 15.7 |
| ALT-LDA | 46.2 | 21.9 |
| ALT-LO | **48.8** | **22.9** |

Table 3.5: Classification accuracy with topic-demoting methods, L2-Reddit dataset.

ALT-LO. We conjecture that our model is stable to demoting topics because it learns relevant stylistic features, rather than spurious correlations.

| | In-Domain | Out-of-Domain |
|---|---|---|
| ALT-LO | 48.8 | 22.9 |
| +MASK TOP-20 | 38.7 | 21.6 |
| +MASK TOP-50 | 36.2 | 21.5 |
| +MASK TOP-100 | 33.5 | 21.2 |
| +MASK TOP-200 | 31.9 | 20.4 |

Table 3.6: Accuracy on the L2-Reddit dataset; the proposed model (ALT-LO) with different settings of the test sets.

## 3.5 Analysis

We present an analysis of what the models are learning, based on words they attend to for classification. We focus on the L2-Reddit dataset.

Following Pryzant et al. (2018), we generated a lexicon of most attended words by (1) running the model on the test set and saving the attention score for each word; and (2) for each word, computing its average attentional score and selecting the top-$k$ words based on this score.

What emerges from this lexicon (Table 3.7) is a dramatic difference between the top indicative words in the various models. Whereas in the baseline model *all* the most indicative words are proper nouns, the ALT-LO model highlights exclusively function words. The proper nouns in the baseline model are all geographical terms directly associated with the L1s reflected in the L2-Reddit dataset: they are easy giveaways of the authors' L1s, but they are meaningless linguistically. In contrast, the function words highlighted in the ALT-LO model are mostly prepositions and determiners; it is well known that nonnative speakers are challenged by the use of prepositions (in any L2, English included). The distribution of determiners is also a challenge for nonnatives, and the correct usage of *the* in particular is quite hard for learners to master. These challenges are evident from the most indicative words of our model. Observe also that the LO-TOP-50 model is somewhere in the middle: it includes some proper nouns (including geographical terms such as *eu* or *us*) but also several function words. A more detailed analysis of these observations is left for future work.

Recently, there has been a debate on whether attention can be used to explain model decisions (Jain and Wallace, 2019; Serrano and Smith, 2019; Wiegreffe and Pinter, 2019; Pruthi et al., 2020), we thus present additional analysis of our proposed method based on saliency maps (Ding et al., 2019). Saliency maps have been shown to better capture word alignment than attention probabilities in neural machine translation. This method is based on computing the gradient of the probability of the predicted label with respect to each word in the input text and normalizing the gradient to obtain probabilities. We use saliency maps to generate lexicons

similar to the ones generated using attention. As shown in table 3.8, the top indicative words for baseline and LO-TOP-50 follow a similar pattern as the ones obtained with attention scores. In line with results in Table 3.7, salient words for ALT-LO are determiners and prepositions. However, saliency maps also reveal that our proposed approach still attends to some geographical terms that were not demoted by our classifier.

| | |
|---|---|
| **NO-ADV** | sweden france greece finland poland spain greek germany french eu romania polish dutch german spanish swedish netherlands finnish |
| **LO-TOP-50** | eu 's 're 'm ' & uk us because 've am its nt english these usa n't here 'll especially correct pis de within |
| **ALT-LO** | the in to of that a i is and 't as from with by ? on but & they are about at because like was would have you |

Table 3.7: The highest scoring words in lexicons generated using attention scores.

## 3.6 Extensions to Other Tasks

**Detecting Variation based on Gender of the Addressee** As we discussed in Chapter 2 (intra-speaker variation), individuals vary their speech based on social situations as well as the identities of their interlocutor. As a way to uncover gender bias in social media comments, Field and Tsvetkov (2020) measure this variation by formulating a task of predicting the gender of the person a given text comment is addressed to. Again, the main challenge in this work is encouraging the model to focus on text features that are indicative of this variation (and hence bias), rather than artifacts in data that correlate with the gender of the addressee. Using the alternative optimization approach to demote topical features we presented above, they report improved performance as well as find evidence of bias against female-identifying individuals.

**Learning Variation Agnostic Classifiers** Finally, we see this approach also shows promise in classification tasks where language variation may be a confound. One prominent example is hate speech detection where prior work has shown that annotator errors and biases as well as training data imbalance may lead to simple classifiers being biased—falsely predicting African-American English text as toxic or offensive (Sap et al., 2019). However, simply amending the goal of the classifier to predict the label and demote the confounding variable (a binary value of race in this case), Xia et al. (2020) achieve an improvement in generalization and fairness across different varieties.

## 3.7 Related Work

**Controlling for confounds in text** Controlling for confounds is an active field of research, especially in the medical domain, where the common solution is to do random trials or propensity score matching (Rosenbaum and Rubin, 1985). Paul (2017) tackled the problem of learning causal associations between word features and class labels using propensity matching for the task of sentiment analysis. This method is not scalable to large text datasets as it involves training a logistic regression model for every word type. Tan et al. (2014) built models to estimate the number of retweets of Twitter messages and addressed confounding factors by matching tweets of the same author and topic. Reis and Culotta (2018) proposed a statistical technique called Pearl's back-door adjustment for text classification (Pearl, 2009). All these works focused on a bag-of-words model with lexical features only. This field of research also has alignment with causal inference and its intersection

with NLP is a growing area of research (Chandrasekharan et al., 2017; Roberts et al., 2020; Egami et al., 2018; Veitch et al., 2020; Keith et al., 2020).

**Adversarial training in text** Much recent work focuses on learning textual representations that are invariant to selective properties of the text. This work used domain adaptation and transfer learning (Ganin et al., 2016; Tzeng et al., 2014; Xie et al., 2017), either to remove sensitive attributes such as demographic information (Beutel et al., 2017; Coavoux et al., 2018), or to understand consumer behavior for social science applications (Pryzant et al., 2018). Most of the work in this area, however, focuses on cases where these confounds are known in advance and their values are given along with the training data. Our presented approach is most closely related to Coavoux et al. (2018) who proposed an alternating optimization method to learn privacy-preserving text representations. This work focuses on demoting binary-valued attributes by maximizing the likelihood of erroneous label using a single adversary network, which we have shown to be inadequate in our experiments. In constrast, we propose a more general method focusing on multinomial distributions which we push towards a uniform distribution with the help of multiple adversaries.

**Native language identification** The L1ID task was introduced by Koppel et al. (2005), who worked on the International Corpus of Learner English (Granger, 2003). The same experimental setup was adopted by several other authors (Tsur and Rappoport, 2007; Wong and Dras, 2009, 2011). Since the release of nonnative *TOEFL* essays by the Educational Testing Service (Blanchard et al., 2013), the task gained popularity and this dataset has been used for two L1ID Shared Tasks (Tetreault et al., 2013; Malmasi et al., 2017).

Malmasi and Dras (2018) report that the state of the art is a linear classifier with character $n$-grams and lexical and morphosyntactic features.

| | |
|---|---|
| **NO-ADV** | poland greek romania greece france spain french sweden finland polish dutch spanish netherlands finnish german |
| **LO-TOP-50** | on 're even 'd up less things 'll doesn living majority sense talk level 've rights took number north |
| **ALT-LO** | the of to i a in greece romania france finland that for is french & you 't finnish |

Table 3.8: The highest scoring words in lexicons generated using saliency maps.

The best accuracy under cross-validation on the TOEFL17 dataset, which includes 11 native languages (with a rather diverse distribution of language families), was 85.2%.

The above works all identify the L1 of *learners*. Identifying the native language of advanced, fluent speakers is a much harder task. Goldin et al. (2018) addressed this task, using the L2-Reddit dataset with as many as 23 different L1s, all of them European and many which are typologically close, which makes the task even harder. They experimented with a variety of features, using logistic regression as the classifier, and achieved results as high as 69% accuracy with cross-validation; however, when testing their classifier outside the domain it was trained on (Reddit forums focusing on European issues), accuracy dropped to 36%.

## 3.8 Conclusions and Future Work

We introduced a method to represent unknown confounds in text classification using topic models and log-odds scores and a method with alternating optimization to learn textual representations which are confound invariant.

We evaluated the proposed solution on the task of native language identification and showed that it learns to make predictions using stylistic features, rather than focusing on topical information. The learning procedure we presented is general and applicable to other tasks that require learning invariant representations with respect to some attribute of text as shown in follow-up works in §3.6. While our results suggest that adversarial training does help reduce the influence of confounding variables, we find in our analysis that it does not eliminate it completely. We identify the reasons for this gap and areas of future research.

First, we construct our weak classifier based on log-odds scores to construct a latent confound representation of topics. We assume here that a weak classifier only uses topical features to make predictions that may not always hold. On the other hand, our experiments with LDA-based topic models did not yield promising results. Future work may investigate different methods of constructing weak classifiers or topic models based on recently published studies to improve upon these issues (Zhao et al., 2021a). Second, we qualitatively analyze our model predictions using attention and saliency maps to understand the behavior of trained models. The reliability of these methods, however, has been debated in the literature (Jain and Wallace, 2019; Serrano and Smith, 2019; Wiegreffe and Pinter, 2019; Pruthi et al., 2020). Future work may benefit from more advanced interpretability methods (LYU et al., 2022) such as those based on training data influence (Koh and Liang, 2017; Han and Tsvetkov, 2021) or explanations or reasoning chains (Wiegreffe et al., 2021; Wei et al., 2022b; Turpin et al., 2023) in understanding the model behavior. Designing interpretability-focused classifiers that demote confounds might also be an avenue for future work (Rajagopal et al., 2021; Ahia et al., 2023). Furthermore, the work presented in this chapter either relies on proxy labels or human annotations to construct datasets. This process itself may introduce additional confounds. Research in this area would benefit from the development of additional evaluation metrics and data sets, such as annotated data that includes information about the context and relevant social variables and modeling approaches that gracefully handle annotator disagreement (as we briefly discuss in the next chapter).

Finally, recent advances in language model pretraining, zero and few-shot learning, and instruction tuning (Brown et al., 2020c; Wei et al., 2022a) have led to rapid improvements in text classification performance for many tasks in many languages. A natural question to ask here is if the issue of spurious correlations is still relevant and worth investigating. Recent work has indicated that it is indeed the case since pretraining data itself is unbalanced along many dimensions and may introduce several biases into the models (Si et al., 2023b; Feng et al., 2023). In such cases, biases can also creep in at the few-shot learning or finetuning stage where having a well-balanced dataset may not be sufficient to create an unbiased model (Si et al., 2023a).

# Chapter 4

# A Multivariate Generative Prompting Framework for Zero-shot Contextualized Text Classification

*This chapter discusses work from Kumar et al. (2023b).*

To understand the nuances of communication, mere text is not enough. Contextual information—who said what to whom, where, and when—can alter the meaning (Eckert, 2012). For example, the same intention to be polite can lead to different utterances in different cultures (Hershcovich et al., 2022). German, Russian, and Polish native speakers tend to use a high level of directness in their requests, which might not be considered polite in English (Ogiermann, 2009; House and Kasper, 2011; Wierzbicka, 2020). Similarly, the same sentiment can be expressed in various ways and the same expression might carry distinct polarities for different people. For instance, the description of a restaurant as "expensive" may indicate a positive polarity for an affluent user although it is generally associated with a negative sentiment (Wang et al., 2018). Other examples include phenomena such as sarcasm, formality, condescension, and empowerment which are often associated with different communicative norms across individuals and cultures (Joshi et al., 2016; Ringel et al., 2019; Wang and Potts, 2019). In NLP applications, numerous studies have highlighted that contextual and socio-demographic information, including text domains, subjects of discussion, communicative goals, author information, and intended audience, can significantly influence what label the text will be assigned (Flek, 2020), especially for subjective classification tasks (Volkova et al., 2013b; Hovy, 2015a; Long et al., 2017). To that end, in this chapter, we explore contextualizing text classification models with extra-linguistic information.

Large training datasets annotated with such information are rarely available and are generally hard to collect due to ethics and privacy risks (Weidinger et al., 2022b). We thus study a zero-shot setting to contextualize text classification tasks using pretrained language models. Extensive prior work has demonstrated that language models can be prompted to solve NLP tasks (Wei et al., 2022a; Liu et al., 2023). Most previous studies have focused on a discriminative classification setup, where the label is predicted by computing its probability conditioned on the input and a prompt. It has also been shown that these setups can be sensitive to variations in the prompt, that is simply paraphrasing the prompt can drastically change the labels (Zhao et al., 2021b; Lu et al., 2022). In recent work, Min et al. (2022) showed that generative prompting (or noisy channel

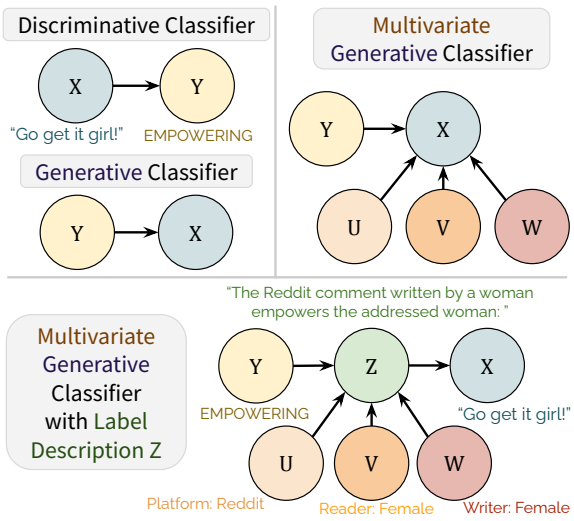



Figure 4.1: Illustration of the proposed multivariate generative classifier with label description and example.

classification), which involves estimating the probability of generating the input given different labels, yields greater stability and better worst-case performance. In this work, we propose to enhance this framework by incorporating contextual information in text classification via expressive label descriptions and propose methods to reduce variance in model performance.

We introduce multivariate generative prompting (Figure 4.1). We first create a description for each label that captures various factors that can influence the label and then make predictions by estimating the likelihood of generating the input text given the label description. This framework enables us to examine and compare the effects of different variables in each text classification task. Furthermore, to reduce variance from different label description variations, we propose to compute and aggregate the results across multiple paraphrases of the label description. For three open-source language model families (GPT2, OPT, Pythia) with models ranging from 125M to 7B parameters, our experiments involving 14 text classification tasks, incorporating additional variables like text source, domain, author, audience, addressee, and the subject of the text, show that our generative prompting setup attains substantial improvements compared to both simple discriminative and single variable generative prompting baselines.

## 4.1 Background

**Language model prompting** Zero-shot and few-shot learning are the two most standard approaches for prompting language models and commonly used for benchmarking their performance on NLP tasks. Zero-shot learning is simply feeding the task text and a prompt to the model and asking for an answer, which we explore this work. Few-shot learning, on the other hand, presents a set of demonstrations, each consisting of both input and desired output, on the target task. As the model first sees good examples, it can better be steered towards the users' intention and criteria for what kinds of answers are wanted (see Table 4.1). Therefore, few-shot learning often leads to better performance than zero-shot. However, it comes at the cost of more token consumption and may hit the context length limit when the input and output text are long. Moreover, the choice of demonstrations can lead to high variance in the model performance (Zhao et al., 2021b) and prior work has investigated various demonstration selection- and ordering strategies to boost performance (Lu, 2022).

| |
|---|
| **(Zero Shot)** Text: i'll bet the video game is a lot more fun than the film.<br>Sentiment: |
| **(Instruction)** Please label the sentiment towards the movie of the given movie review. The sentiment label should be "positive" or "negative".<br>Text: i'll bet the video game is a lot more fun than the film.<br>Sentiment: |
| **(Few Shot)** Text: (lawrence bounces) all over the stage, dancing, running, sweating, mopping his face and generally displaying the wacky talent that brought him fame in the first place.<br>Sentiment: positive<br>Text: despite all evidence to the contrary, this clunker has somehow managed to pose as an actual feature movie, the kind that charges full admission and gets hyped on tv and purports to amuse small children and ostensible adults.<br>Sentiment: negative<br>Text: for the first time in years, de niro digs deep emotionally, perhaps because he's been stirred by the powerful work of his co-stars.<br>Sentiment: positive<br>Text: i'll bet the video game is a lot more fun than the film.<br>Sentiment: |

Table 4.1: Illustration of zero-shot, instruction-based and few-shot discriminative classification from language models for binary sentiment classification. The final prediction can be made by computing the probability of the words 'positive' and 'negative' given the contexts and predicting the label with the highest probability.

Intuitively, the purpose of few-shot examples in the prompt is to explain users' intent to the model; in other words, describe the task instruction to the model in the form of demonstrations. Recent work has also studied providing such instructions or prompts, either to a pretrained language model directly or finetuning a model to follow instructions using a collection of NLP tasks (Wei et al., 2022a). Instructions and few-shot learning can also be used together. Again, depending on how instructions are phrased, however, can significantly alter the model outputs, even in instruction fine-tuned models (Sun et al., 2023). In contrast, several studies have also developed prompt engineering techniques, that is creating a sequence of prefix tokens or prompts that increase the probability of getting desired output given input (Liu et al., 2023). These techniques rely on available training data for each task. In this work, we focus on a zero-shot prompting setup operating in a setting where no training data for customizing classification models is available.

**Discriminative versus Generative Classification** Text classification studies with prompting have primarily focused on discriminative classification, which focus on constructing an input prompts that get prepended to each input text to predict the classification label. That is conditioning on the input to generate the output.

Generative or noisy channel models (Brown et al., 1993) have been previously investigated for various NLP tasks, such as machine translation (Yamada and Knight, 2001; Yee et al., 2019) and question answering (Lewis and Fan, 2019). Prior work has empirically demonstrated that generative models are more robust to distribution shift in text classification than discriminative models (Yogatama et al., 2017a). Recently, Min et al. (2022) explored the use of a generative model with prompting, leveraging pretrained language models for various text classification tasks. In this work, we extend the generative model with text prompts into a multivariate framework by incorporating label descriptions. These descriptions capture various contextual information associated with each example, allowing for effective priming and customization of the classifier.

**Social and personal factors in NLP** Machine learning systems have been shown to reflect and amplify social prejudices in human-written text, resulting in systemic bias in performance towards specific demographic groups (Mehrabi et al., 2021). As we discussed in the last chapter, such classifiers learn spurious correlations between the label and the demographic information reflected in text either explicitly through their mentions in the text (such as names, sexuality, and race among others) or their writing style. These issues are exacerbated through annotation artifacts (Sap et al., 2019, 2022b) or unbalanced datasets (Kiritchenko and Mohammad, 2018). Various solutions proposed in the literature aim to learn models that are fair to all demographics using methods like adversarial learning (Han et al., 2021a,b) and distributionally robust optimization (Michel et al., 2021; Zhou et al., 2021). A distinct but closely related motivation towards developing such solutions is user privacy—models should never use any personally identifiable attributes to make any predictions as it could lead to unintended negative consequences (Elazar and Goldberg, 2018). Ravfogel et al. (2020, 2022) propose methods to scrub demographic information from model representations given a trained model with little loss in model accuracy.

In contrast, few studies have shown that incorporating factors such as gender, age, region, or country of the authors as features can improve text classification performance (Volkova et al., 2013a; Hovy, 2015b; Yang and Eisenstein, 2017; Lynn et al., 2017; Huang and Paul, 2019). Most of these studies are based on the assumption that social and personal factors are causally related to both the writing style and the target label. As a result, they treat classification as a domain adaptation problem in which demographic attributes divide the data distribution into different domains. In this chapter, we are also interested in exploring the impact of extra-linguistic features in text classification with one crucial difference. We operate under the setting that social and personal factors are not reflected in the writing itself but can contextualize or disambiguate the predictions providing information not necessarily reflected in the input text. That is the same text variation can have potentially multiple originating sources and need to be specified to make accurate predictions. The most closely related work to our work is personalized classifiers which operate at the level of idiolects (Mireshghallah et al., 2022b).

**Cross-Lingual Transfer via Cultural Similarities or Differences** Many cultures span across speakers of multiple regions and languages. Sun et al. (2021) develop a distance measure based on cultural similarities across speakers of different languages and show that training multilingual models on culturally close languages can offer substantial improvements in sentiment classification compared to using typologically similar languages. Ringel et al. (2019) on the other hand exploits known cultural differences between speakers of two languages as distant supervision, e.g. they show that training a classifier to distinguish human written English text from those translated from German to English can be used to detect formality in English. In this work, we are interested in subcultures within a language, where the same text may be assigned a different label based on speaker and listener identities and their relationships (Radfar et al., 2020; Danescu-Niculescu-Mizil et al., 2013).

## 4.2 Multivariate Generative Classifier with Label Descriptions

Our goal is to design a function $f: \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots\}$ is a set of natural language texts and $\mathcal{Y} = \{y_1, \cdots, y_n\}$ is a set of classes. Traditionally, $f$ is instantiated as a discriminative classifier which models $\hat{y} = \arg\max_{y_i \in \mathcal{Y}} p(y_i|\mathbf{x})$. These classifiers take as input the text $\mathbf{x}$ and output a label distribution. That is, they are designed to distinguish between the correct label and other possible label choices. An alternative

formulation proposed in the literature is a generative classification framework that reinterprets this objective using Bayes' rule and a different factorization as,

$$\begin{aligned} p(y_i|\mathbf{x}) &\propto p(\mathbf{x}, y_i)/p(\mathbf{x}) \\ &= p(\mathbf{x}|y_i)p(y_i) \end{aligned}$$

Assuming a balanced label setup, $p(y_i)$ can be ignored. In addition, $p(\mathbf{x})$ does not depend on the label. Hence at inference time, the classification objective becomes,

$$\hat{y} = \arg\max_{y_i \in \mathcal{Y}} p(\mathbf{x}|y_i)$$

In this setup, we assume a label is generated first (e.g., an author decides to write a negative review), and then the text (e.g., the negative review) is produced conditioned on the label. Prior work hypothesizes that generative models can be more accurate than discriminative models which may look for shortcuts to predict the label (Yogatama et al., 2017a).

In this work, we extend this setup to multivariate generative classification, to generalize to more variables that might influence the generative process of the input text, expressing the generative probability of $\mathbf{x}$ as $p(\mathbf{x}|y, u, v, \ldots)$, where $u$, $v$, ... represent the additional factors. For example, to generate a review, not only is the author influenced by the polarity but also by the item they review, the medium where they write the review, their target audience, and their writing style. In this work, we are interested in measuring the effects of semantically defined variables on classification performance.

Specifically, we focus on a zero-shot inference setup from autoregressive language models trained on raw text and not finetuned on any supervised data.[1] As introduced in Brown et al. (2020b), language models can be used in a zero-shot setup by computing $p(z(y_i)|x)$ or in our case, $p(x|z(y_i, u, v, \ldots))$. Here, $z(\cdot)$ is often called a verbalizer which expresses the label in natural language form so that meaningful probabilities can be computed. In this work, since the verbalizer is only concerned with the label, we refer to it as a *label description*. A simple example is "This is terrible." and "This is amazing." for negative and positive labels respectively. The choice of this description, however, can lead to large variance in the model performance (Liu et al., 2023; Holtzman et al., 2021; Zhao et al., 2021b).

To reduce this variance, we propose to use multiple variations of prompts $z$. More formally, the labels and other variables generate a label description which then informs the generation of the text,

$$\begin{aligned} p(y_i|\mathbf{x}, u, v, \ldots) &\propto p(\mathbf{x}, y_i|u, v, \ldots) \\ &= \sum_{z \in \mathcal{Z}(y_i, u, v, \ldots)} p(\mathbf{x}, y_i, z|u, v, \ldots) \\ &= \sum_{z \in \mathcal{Z}(y_i, u, v, \ldots)} p(\mathbf{x}|y_i, z, u, v, \ldots)p(z|y_i, u, v, \ldots)p(y_i)p(u)p(v)\ldots \end{aligned}$$

$p(u), p(v), \ldots$ are independent of the label $y_i$ and can be dropped. Further, assuming equal likelihood of each label[2] *and* equal likelihood for each description given the label and other variables, we drop $p(y_i)$ and $p(z|y_i, u, v, \ldots)$. Given the description $z$, the input text $\mathbf{x}$ is independent of the other variables (see Figure 4.1 bottom). Hence, the first term in the summation can be reduced to $\sum_z p(x|z)$, which is our inference objective.

[1] Future work may study this in a few-shot/finetuning setup.

[2] This is a simplifying assumption we may which may not always be true. Future work may study modeling the priors $p(y), p(u), \ldots$ especially where certain contextual factors may have higher correlations with certain labels.

| Task | Label Description |
|---|---|
| Sentiment | "This [DOMAIN] leans [POLARITY]: "; DOMAIN∈{text, movie review, Yelp review, poem verse, financial news excerpt}, POLARITY∈{very positive, positive, neutral, negative, very negative}. We use "positive" and "negative" as POLARITY for binary sentiment classification. |
| Hate speech | "This [DOMAIN] uses [LABEL] language: "; DOMAIN∈{text, reddit post}, LABEL∈{hateful, innocuous}. |
| Ethos | "This [DOMAIN] contains hate-speech about [SUBJECT]: "; DOMAIN∈{text, social-media comment}, SUBJECT∈{something, national origin, religion, race, sexual orientation}. This is a binary classification task where "something" serves as the negative class for every other subject. |
| Topic | "The topic of this [DOMAIN] revolves around [TOPIC]: "; DOMAIN∈{text, news excerpt}, TOPIC∈{world, sports, business, science and technology} |
| Politeness | "According to a [AGE] years old person with a [EDUCATION], this email snippet is impolite:"; AGE∈ set of integers, EDUCATION∈{High school degree, college degree, }. Both are provided in the test example. |
| Emotion | "This [DOMAIN] emotes [EMOTION]: "; DOMAIN={text, tweet}, EMOTION={sadness, love, anger, joy, fear, surprise} |
| Empowerment | "This Reddit comment written by a [AUTHOR] empowers and uplifts the addressed [ADDRESSEE]: "; AUTHOR={man, woman}, ADDRESSEE={man, woman} |

Table 4.2: Label description starter templates we hand write. DOMAIN="text" represents missing domain information. We generate their variations by asking ChatGPT: "Write 10 paraphrases of this sentence as a Python list."

That is, we break down the generative classification objective into a sum of multiple objectives where each objective considers a variation of the label description. We also compare with other aggregation strategies in ablation studies. We compute this term for each label under consideration $y_i$ and predict the label which obtains the highest value. Notably, unlike common prompting scenarios, the label descriptions, $z$, are unique for each label being considered and can be specialized by adding any available information about the instance in natural language format.

## 4.3 Experimental Setup

**Datasets and Models** We report results on 14 text classification datasets encompassing diverse tasks, domains, and difficulty levels. These datasets include varying numbers of classes and attributes that can be used as additional input to improve the classification performance. We consider the following tasks divided in to two groups: (1) Sentiment, Topic, and Hate Speech which in addition to text are accompanied by information about the domain, source or subject of the input text, and (2) Politeness, and Empowerment which includes demographic information of the author, addressee, or the reader. Table 4.3 summarizes the details of each dataset we use. We only focus on zero-shot performance using publicly available validation or test sets, without using the training data at all. We experiment with the three classes of open-source models: GPT2 (Small, Medium, Large, and XL) (Radford et al., 2019a), OPT (Zhang et al., 2022)(1.4B and 3B), and Pythia (Biderman et al., 2023) (1.4B, 2.8B and 6.7B).

**Label Descriptions** For each task, we manually write one label description per label using a template (see complete list in Table 4.2). We then generate 10 paraphrases of each label description by querying ChatGPT.[3]

[3]We used the free tier of ChatGPT for this purpose: https://chat.openai.com/chat.

| Dataset | Task (Domain) | #Classes |
|---|---|---|
| SST-2 (Socher et al., 2013) | Sentiment Classification (movie) | 2 |
| SST-5 (Socher et al., 2013) | Sentiment Classification (movie) | 5 |
| Yelp (Zhang et al., 2015) | Sentiment Classification (Yelp) | 5 |
| Poem Sentiment (Sheng and Uthus, 2020) | Sentiment Classification (Poetry) | 4 |
| Financial Phrasebank (Malo et al., 2014) | Sentiment Classification (Economic News) | 3 |
| Hate_speech18 (de Gibert et al., 2018) | Hate Speech (Reddit) | 2 |
| Ethos (4 subsets) (Mollas et al., 2022) | Hate Speech by Subject (various social media) | 2 |
| Emotion (Saravia et al., 2018) | Emotion Recognition (Twitter) | 6 |
| Potato Prolific (Pei and Jurgens, 2023) | Politeness Classification (Email) | 2 |
| Talk Up (Njoo et al., 2023) | Empowerment prediction (Reddit) | 2 |

Table 4.3: Datasets used for the experiments

This process needs to be done only once for each task and, in practice, any paraphrasing model can be employed.

**Baselines** We primarily compare our proposed approach with the following zero-shot baselines. Additionally, we include comparisons with previously reported demonstration-based few-shot results, although they are not our direct baselines.

- **DISC** predicts the label using $p(y_i|\mathbf{x})$ where $y_i$ is described using the same label descriptions. To condition on additional variables, we prepend a description to $\mathbf{x}$.
- **DISC-PMI** uses $p(y_i|\mathbf{x})/p(y_i|\text{NULL})$ for inference. Since language models probabilities can be poorly calibrated and suffer from competition between different surface forms with the same meaning, this method relies on pointwise mutual information (PMI) between $\mathbf{x}$ and $y$ to make a prediction (Holtzman et al., 2021).

## 4.4 Results

We categorize the tasks into two groups: domain-aware classification, which considers the domain of the text as an additional factor, and personalized classification, which includes personal attributes of writers and readers as additional factors.

**Domain-Aware Classification** Table 4.4 shows the performance of the different methods on sentiment, topic, emotion, and hate speech classification for GPT2-Large. Remaining results are reported in Figure 4.2. Table 4.5 shows comparisons of a subset of datasets with previously reported few-shot methods (Min et al., 2022; Wang et al., 2023).

We find that our proposed approach substantially outperforms discriminative approaches in the zero-shot setting. Among the three methods evaluated, DISC consistently performs the poorest, often close to random performance, while DISC-PMI shows improvement by mitigating surface form competition. Our zero-shot method also either outperforms or matches the strong few-shot baselines (Table 4.5). Notably, our method shows minimal variance in performance due to prompt selection by aggregating over multiple prompt paraphrases, whereas few-shot baselines exhibit large deviations (up to 6.1%).

We conduct ablation studies to assess the impact of each proposed component on performance.

| | DISC | DISC-PMI | Ours |
|---|---|---|---|
| SST-2 | $56.5_{(0.48)}$ | $76.8_{(0.46)}$ | $\mathbf{86.9}_{(0.21)}$ |
| SST-5 | $21.3_{(0.26)}$ | $25.6_{(0.0)}$ | $\mathbf{34.5}_{(0.39)}$ |
| Yelp | $29.8_{(0.07)}$ | $39.2_{(0.04)}$ | $\mathbf{42.8}_{(0.05)}$ |
| PS | $29.9_{(3.11)}$ | $48.8_{(2.48)}$ | $\mathbf{58.5}_{(1.96)}$ |
| FP | $24.7_{(0.30)}$ | $47.2_{(0.70)}$ | $\mathbf{60.2}_{(0.65)}$ |
| AGNews | $53.9_{(0.16)}$ | $68.4_{(0.30)}$ | $\mathbf{70.9}_{(0.14)}$ |
| Emotions | $35.8_{(0.33)}$ | $\mathbf{40.5}_{(0.42)}$ | $39.0_{(0.28)}$ |
| Hate_Speech18 | $11.4_{(0.07)}$ | $38.9_{(0.39)}$ | $\mathbf{62.9}_{(0.71)}$ |
| Ethos (NO) | $44.6_{(1.61)}$ | $66.7_{(1.63)}$ | $\mathbf{80.2}_{(0.86)}$ |
| Ethos (SO) | $25.9_{(1.38)}$ | $67.4_{(3.05)}$ | $\mathbf{70.3}_{(2.04)}$ |
| Ethos (Race) | $18.4_{(0.00)}$ | $38.4_{(1.64)}$ | $\mathbf{63.3}_{(2.08)}$ |
| Ethos (Religion) | $31.8_{(2.91)}$ | $70.9_{(1.16)}$ | $\mathbf{77.6}_{(3.26)}$ |

Table 4.4: Zero-shot accuracy with GPT2-Large. We use 10 label descriptions for each class. We report $\text{average}_{\text{std}}$ over 10 runs with different sets of label descriptions in each run. More results are provided in the appendix. PS: Poem Sentiment, FP: Financial Phrasebank.

| | Few-shot | | | Zero-shot |
|---|---|---|---|---|
| | DISC | DISC-PMI | GEN | Ours |
| SST2 | $58.9_{(9.4)}$ | $79.7_{(5.8)}$ | $85.0_{(1.1)}$ | $\mathbf{86.9}_{(0.21)}$ |
| SST5 | $27.6_{(5.2))}$ | $33.8_{(5.8)}$ | $\mathbf{36.2}_{(2.1)}$ | $34.5_{(0.39)}$ |
| Yelp | $32.6_{(5.1)}$ | $39.2_{(6.1)}$ | $41.5_{(6.1)}$ | $\mathbf{42.8}_{(0.05)}$ |
| AGNews | $51.9_{(9.8)}$ | $73.1_{(6.2)}$ | $\mathbf{74.3}_{(2.7)}$ | $70.9_{(0.14)}$ |

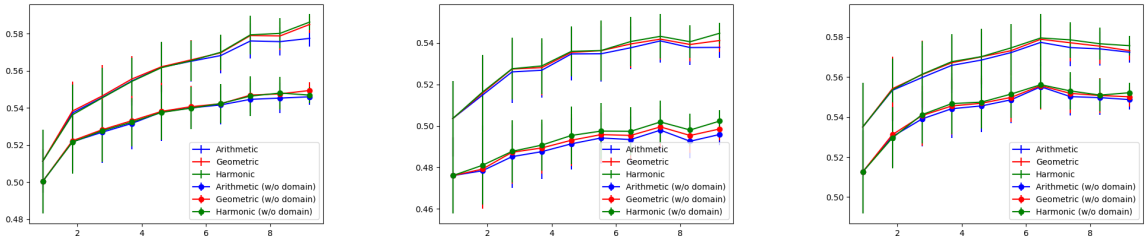
Table 4.5: Our zero-shot versus previously reported few-shot classification results in Min et al. (2022) (GEN is their proposed method).

- **Effect of number of label descriptions.** In this ablation, we vary this number from $k$=1 to $k$=9 and observe the change in performance. For each $k$, we do this evaluation 10 times and report the mean and standard deviation. We find that in the majority of cases, increasing the number of label descriptions improves the model performance highlighting the utility of this approach. However, we observe that the performance starts to stabilize between $k$=5 and $k$=9. In contrast, for discriminative baselines, we observe no clear trend as increasing $k$ sometimes results in a decrease in performance.

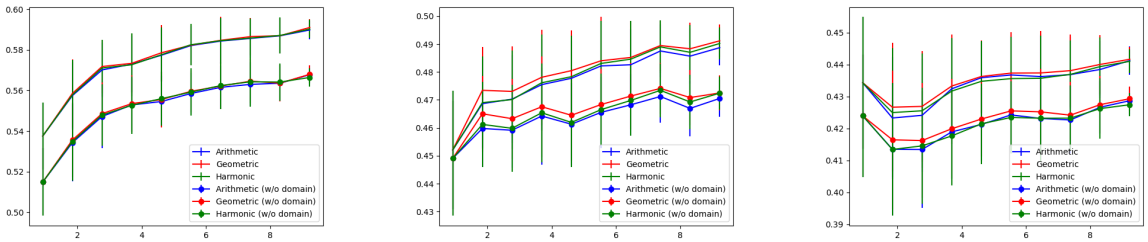
- **Effect of additional variables.** To measure the effect of provided contextual information (domains, subject, or, data source), we conduct ablation studies on all four tasks in Table 4.4, by modifying the label description to exclude this information. We report the full results in Figure 4.3 and Figure 4.4. We observe a significant drop in the performance across all tasks if we remove the domain or data source information including our method as well as the baselines. We hypothesize that specifying the domain information helps prime the model probabilities to the right distributional landscape allowing more meaningful comparisons between probabilities assigned to different labels.

- **Effect of model size.** We measure if the presented methods holds across model scales. We repeat the same experiment across GPT2 (S, M, L, XL) models ranging from 125M to 1B, two large OPT models of size 1.3B and 2.7B parameters, and three Pythia models 1.4B, 2.7B and 6.7B parameters respectively.[4].

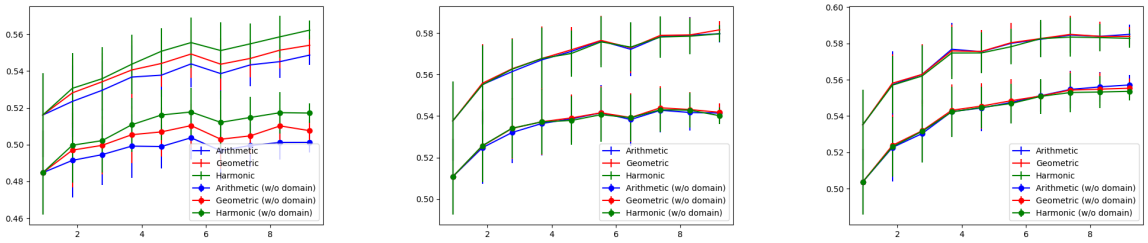
[4]Our computational budget prevents us from experimenting with even larger models. We leave that for future work.




(a) GPT-2 Small (b) GPT2 Medium (c) GPT2 Large




(d) GPT2 XL (e) OPT 1.3B (f) OPT 2.7B




(g) Pythia 1.4B (h) Pythia 2.8B (i) Pythia 6.7B

Figure 4.2: Full results for domain aware classification for our proposed setup. The x-axis shows number of label descriptions per label and the y-axis indicates the average accuracy across all the tasks except (Politeness and Empowerment). We conduct a thorough analysis of this setup as discussed in (section 4.4): removing domain information from the descriptions, different aggregation strategies as well as evaluating on different model sizes and families.

We find that across reasonably large models, going larger improves performance on average. We see substantial improvement from GPT2-M to L to XL, and Pythia 1.4B to 2.7B to 6.7B.[5] We observe opposite trends in OPT models and GPT-2 S to M which are consistent with prior work (Wang et al., 2023) and require further investigation.

- **Ablation on aggregation strategy** We aggregate the probabilities obtained using different label descriptions by simply summing them (same as arithmetic mean, for comparison purposes). This aggregation is theoretically grounded in the probabilistic framework we design (Figure 4.1). Prior work has considered several other aggregation strategies that we compare with in this ablation. We compare against geometric mean (or arithmetic mean of the log probabilities) and harmonic mean. We find that in our proposed generative setup, the performance across three aggregation strategies is largely similar, harmonic mean

[5]Note that across model families, the performance is not strictly comparable due to differences in pretraining corpora.

| Author \ Addressee | No | Yes | Age \ Education | No | Yes |
|---|---|---|---|---|---|
| No | $81.1_{(1.43)}$ | $84.8_{(1.83)}$ | No | $80.1_{(0.30)}$ | $81.4_{(0.21)}$ |
| Yes | $81.8_{(1.07)}$ | $\mathbf{85.0}_{(2.42)}$ | Yes | $82.2_{(0.32)}$ | $\mathbf{83.3}_{(0.24)}$ |

Table 4.6: Personalized classification results (F1-scores) on GPT2-Large with our model. Each cell represents whether the demographic attribute was used in the label description or not.

outperforming the other two slightly. We hypothesize that this effect is due to harmonic mean's property of ignoring outliers. Future work may analyze this strategy in-depth. For the discriminative setup, the picture is unclear.

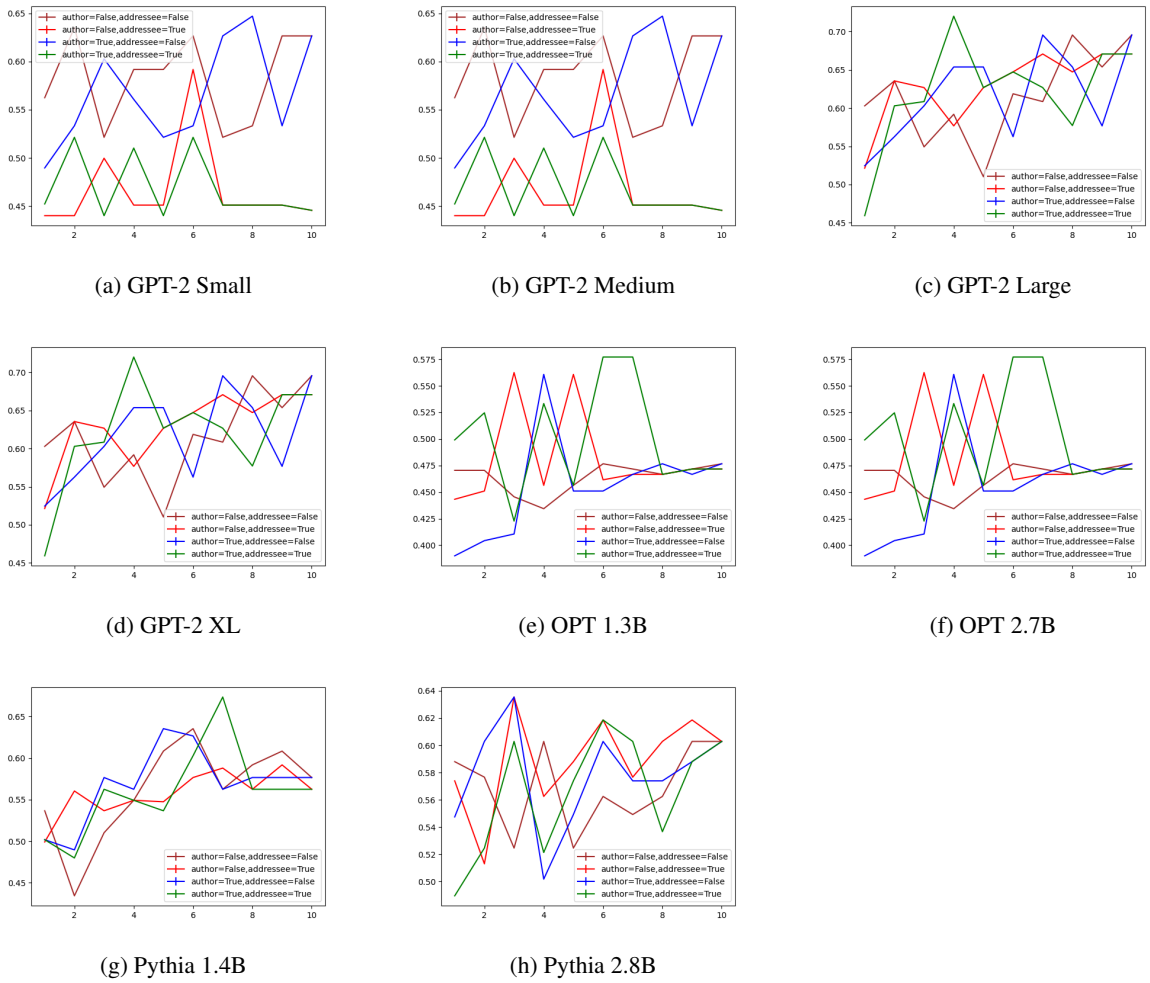
**Personalized Classification** In this setup, we evaluate our proposed approach on two datasets where personal information about the author, the addressee, or even the audience may affect the prediction. We experiment with two tasks: (1) empowerment prediction (Njoo et al., 2023) where given a Reddit comment, the goal is to predict whether it empowers or disempowers the addressee of the comment. We use the author's and the addressee's gender in this task.[6] (2) Politeness prediction (Pei and Jurgens, 2023) where given an email snippet the goal is to predict whether it is polite or not. We again consider binary labels. What is considered polite may vary with the reader dependent cultural factors. This dataset consists of information about the annotator's age, gender, race, and educational background. We focus on age and educational background as they were the primary delineators of variation measured by the authors. That is given the author's age and educational background, we predict the perceived politeness of the text and sum their probabilities to make the final predictions. We do not aggregate these probabilities over each possible value of age and educational background but rather use only the ones reported in the test set. The results for both datasets for GPT2-Large are reported in Table 4.6. The results for other models can be found in Figure 4.3 and Figure 4.4. We report the results for only our proposed approach with varying number of personal attributes considered, as we found the discriminative models to perform poorly in this setup (close to random performance across both tasks). We find that for both test sets, personalizing the predictions with demographic variables helps improve performance. For empowerment prediction, the gender of the addressee, and politeness, the age of the annotator affect the performance more than the other variables. The latter is consistent with prior studies that show cultural differences in politeness across different age groups (Pei and Jurgens, 2023).

## 4.5 Conclusions and Future Work

We present multivariate generative prompting: a text classification framework that uses language model likelihood of generating the input text given different label descriptions to make predictions. We show that incorporating contextual information beyond the text itself into an expressive label description can help improve classification performance as well as personalize predictions. While in this work, we study the setting of discriminative versus generative prompting, future work may combine them both (Raina et al., 2003) to extract further improvements from pretrained LMs. Further, we evaluate this setup only in zero-shot settings with language models trained on raw text, further improvements may be achieved with few-shot learning, instruction-based model fine-tuning or even pretraining models with such attributes (Keskar et al., 2019b).

Now we highlight the limitations of this work and identify areas of future research. First, our operational

[6]We use binary gender here; the evaluation set does not contain any other information




(a) GPT-2 Small (b) GPT-2 Medium (c) GPT-2 Large

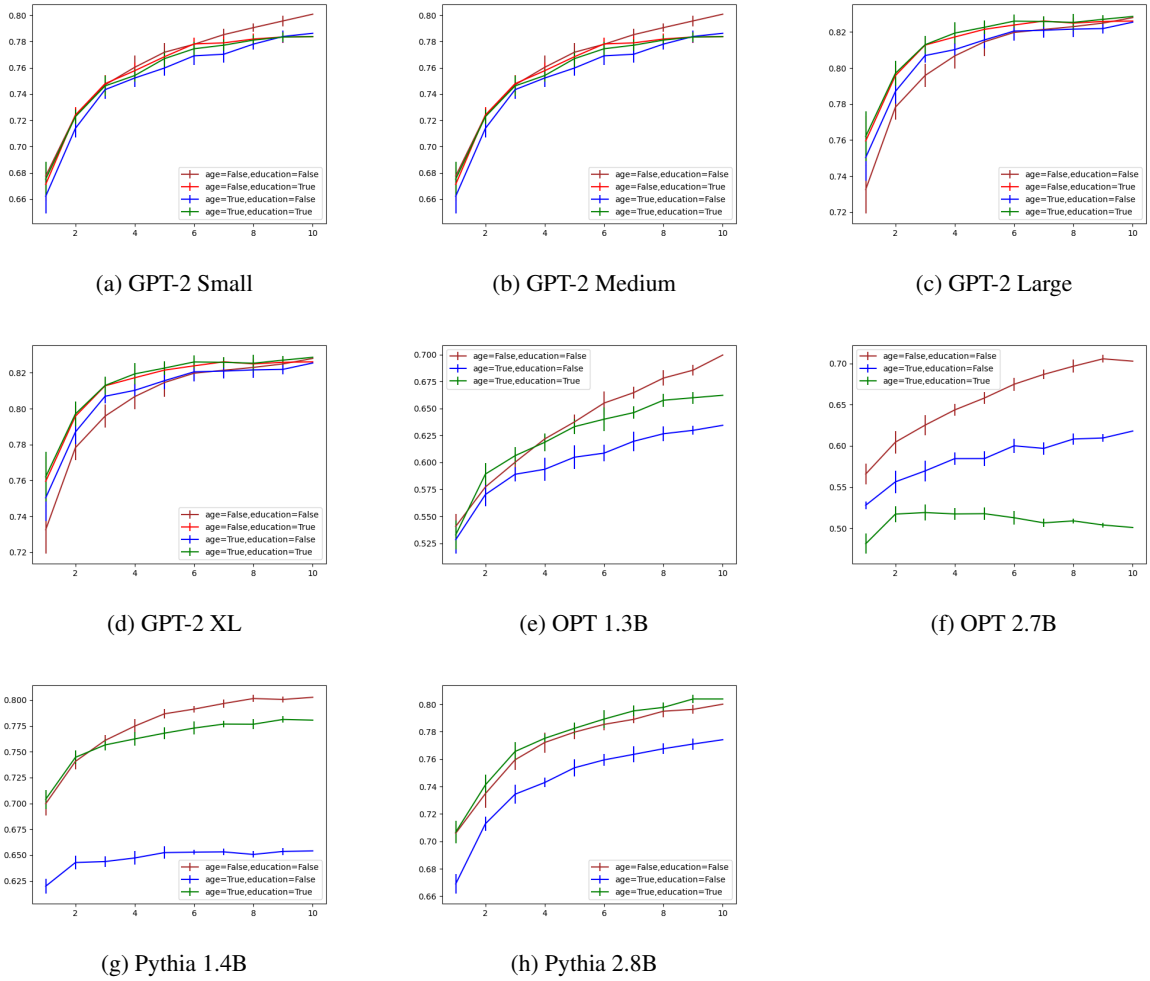
(d) GPT-2 XL (e) OPT 1.3B (f) OPT 2.7B

(g) Pythia 1.4B (h) Pythia 2.8B

Figure 4.3: Full results for author and addressee-personalized empowerment prediction with our proposed setup. The x-axis shows number of label descriptions per label and the y-axis indicates the average F1-score. The four plots indicate 4 settings described in Table 4.6.

assumption that demographic attributes like age, gender and education levels correlate with task labels is population-centric and may not reflect individual preferences. Future work may extend this work to personalize to individuals where few-shot learning may be employed where few-shot example selection can be personalized to individual users (Li et al., 2023). Further, certain demographic attributes in our work may not fully represent the entire population. For example, due to data availability, we only conducted experiments with binary gender. Additionally, the definition and categorization of social attributes in the datasets used in our experiments might predominantly reflect Western-centric perspectives, as the majority of the work involved in designing and creating such datasets aligns with Western-centric viewpoints. Additionally, we exclusively conduct experiments with English datasets. Future work may study the utility of this approach on other languages and cultures (Scao et al., 2022).

Finally, in the experimental setup, we assume that all demographic identifiers are available make predictions. For several reasons, including privacy concerns, this information may not available at test time. For such cases, future work may explore building solutions to recognize ambiguous examples and abstain from making



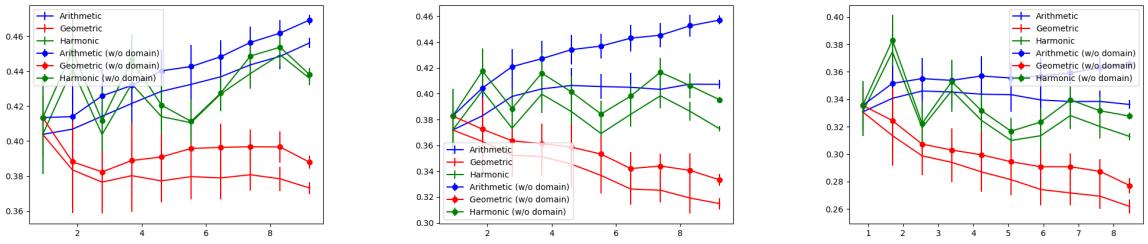

(a) GPT-2 Small (b) GPT-2 Medium (c) GPT-2 Large

(d) GPT-2 XL (e) OPT 1.3B (f) OPT 2.7B

(g) Pythia 1.4B (h) Pythia 2.8B

Figure 4.4: Full results for reader-personalized politeness prediction with our proposed setup. The x-axis shows number of label descriptions per label and the y-axis indicates the average accuracy. The four plots indicate 4 settings described in Table 4.6.

a prediction without appropriate context (Balsubramani, 2015). Keeping with the theme of this thesis, this can also be formulated it as multi-objective optimization problem (Gangrade et al., 2021) where associating abstaining with a cost, the goal is to maximize model accuracy while minimizing this cost.

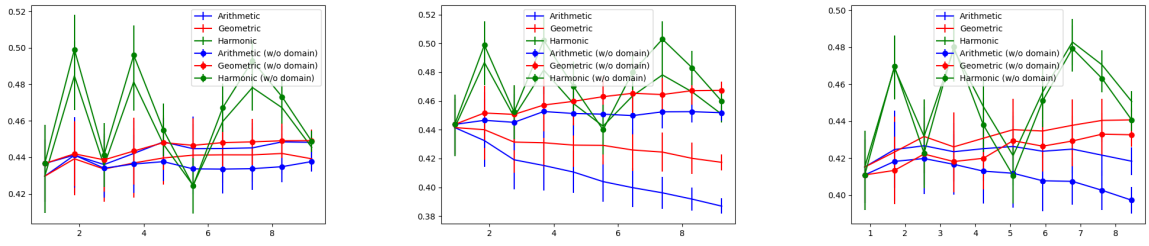



(a) GPT-2 Small (b) GPT2 Medium (c) GPT2 Large

(d) GPT2 XL (e) OPT 1.3B (f) OPT 2.7B

(g) Pythia 1.4B (h) Pythia 2.8B (i) Pythia 6.7B

Figure 4.5: Full results for domain aware classification for DISC. The x-axis shows number of label descriptions per label and the y-axis indicates the average accuracy across all the tasks except (Politeness and Empowerment). We conduct a thorough analysis of this setup as discussed in (section 4.4): removing domain information from the descriptions, different aggregation strategies as well as evaluating on different model sizes and families.




(a) GPT-2 Small (b) GPT2 Medium (c) GPT2 Large

(d) GPT2 XL (e) OPT 1.3B (f) OPT 2.7B

(g) Pythia 1.4B (h) Pythia 2.8B (i) Pythia 6.7B

Figure 4.6: Full results for domain aware classification for DISC-PMI. The x-axis shows number of label descriptions per label and the y-axis indicates the average accuracy across all the tasks except (Politeness and Empowerment). We conduct a thorough analysis of this setup as discussed in (section 4.4): removing domain information from the descriptions, different aggregation strategies as well as evaluating on different model sizes and families.

# Part II

# Language Variation and Text Generation

# Chapter 5

# Training Text Generation Models Adaptable to Language Varieties

*This chapter discusses work previously published in Kumar and Tsvetkov (2019), Bhat et al. (2019), Jegadeesan et al. (2021), and Kumar et al. (2021a).*

Due to the power law distribution of word frequencies, rare words are extremely common in any language (Zipf, 1935). Yet, the majority of language generation tasks—including machine translation (Sutskever et al., 2014; Bahdanau et al., 2015a; Luong et al., 2015), summarization (Rush et al., 2015; See et al., 2017), dialogue generation (Vinyals and Le, 2015), question answering (Yin et al., 2016), speech recognition (Graves et al., 2013; Xiong et al., 2017), and others—generate output tokens by searching or sampling from a multinomial distribution over a *fixed* vocabulary generated using a softmax layer.

Traditionally, this vocabulary is defined by segmenting text into linguistically motivated words. For computational feasibility, the output vocabulary is limited to a few tens of thousands of most frequent words, sacrificing linguistic diversity by replacing the long tail of rare words with an unknown word token, UNK. Further, once the models are trained, this vocabulary cannot be easily modified without retraining the model. More recently, several frequency-based tokenization approaches have been proposed (Sennrich et al., 2016; Schuster and Nakajima, 2012; Kudo, 2018; Kudo and Richardson, 2018) which split words in smaller character n-grams or *sub*words—reducing the effective vocabulary size by segmenting words in the long tail into smaller units making the models open vocabulary, in principle. These tokenizers, however, are prone to excessive fragmentation of rarer or unseen words, such as those in different dialects and domains, leading to inferior performance when generating them (Chronopoulou et al., 2020; Wang et al., 2021b; Tay et al., 2022). This operation is still computationally slow; it follows a large matrix multiplication to compute scores over the candidate tokens. This can make it expensive in terms of memory requirements and the number of parameters to learn (Morin and Bengio, 2005; Mnih and Kavukcuoglu, 2013; de Brébisson and Vincent, 2016). Recent work has also explored defining tokens as characters or bytes, but it can make the tokenized sequences extremely long in terms of the number of tokens.[1]

Towards addressing these issues, this chapter introduces an alternative foundational approach to training text generation models which allows easy adaptability to novel lexical items post-training by treating generation as a step-wise regression problem, as opposed to the widely used notion of treating it as step-wise classification (via softmax). We propose to separate lexical representation learning from model learning: we represent

[1] See Mielke et al. (2021) for a detailed survey on tokenization.

each token as a low-dimensional continuous vector, and train the generation model to predict these vectors at each decoding step instead of a probability distribution over the vocabulary. We use as a training objective the distance between the output vector and target lexical representations. To avoid degenerate solutions, this objective has to be constrained such that no two token representations should be equal. Softmax formulation naturally satisfies this constraint but it is computationally slow. In this work, we instead pre-train (Mikolov et al., 2013; Bojanowski et al., 2017a) and fix the token representation using an auxiliary objective to prevent collapse (this objective can incorporate information from subwords or characters of the token learning richer representations). At test time, the model generates a vector and then searches for its nearest neighbor in the target vector space to generate the corresponding token.

While this idea is simple and intuitive, in practice, we find it does not yield competitive performance with standard regression losses like $\ell_2$. This is because $\ell_2$ loss implicitly assumes a Gaussian distribution of the output space which is likely false for embeddings. In order to correctly predict the outputs corresponding to new inputs, we explore an alternative probability distribution of the target vector conditioned on the input (Bishop, 1994). A major contribution of this chapter is a new loss function based on defining such a probability distribution over the word embedding space and minimizing its negative log likelihood (§5.1.1).

We first present the details of this training method in §5.1 with experiments in machine translation on datasets with huge vocabulary sizes (up to 500K). We show that our proposed approach trains up to 2.5x faster than softmax-based models (based on recurrent architectures) while performing on par with them in terms of generation quality. Error analysis reveals that the models with continuous outputs are better at correctly generating rarer words than the baselines. In §5.2, we present a simple and effective approach to adapt these trained models to generate dialects of the target language with a different but related target vocabulary in extremely low-resource scenarios.

## 5.1 Background: Language Generation with Continuous Outputs

Traditionally, language generation models use one-hot representations for each word in the output vocabulary $\mathcal{V}$. More formally, each word $w$ is represented as a unique vector $\mathbf{o}(w) \in \{0,1\}^V$, where $V$ is the size of the output vocabulary and only one entry $id(w)$ (corresponding the word ID of $w$ in the vocabulary) in $\mathbf{o}(w)$ is 1 and the rest are set to 0. The models produce a distribution $\mathbf{p}_t$ over the output vocabulary at every step $t$ using the softmax function:

$$\mathbf{p}_t(w) = \frac{e^{s_w}}{\sum_{v \in \mathcal{V}} e^{s_v}} \tag{5.1}$$

where, $s_w = W_{hw}\mathbf{h}_t + b_w$ is the score of the word $w$ given the hidden state $\mathbf{h}$ produced by the model at time step $t$. $W \in \mathbb{R}^{V \text{x} H}$ and $b \in \mathbb{R}^v$ are trainable parameters. $H$ is the size of the hidden layer $\mathbf{h}$.

These parameters are trained by minimizing the negative log-likelihood (i.e. cross-entropy) of this distribution by treating $\mathbf{o}(w)$ as the target distribution. The loss function is defined as:

$$\text{NLL}(\mathbf{p_t}, \mathbf{o}(w)) = -\log(\mathbf{p_t}(w))$$

This loss computation involves a normalization proportional to the size of the output vocabulary $|\mathcal{V}|$. This can be a bottleneck when the vocabulary is large. We instead propose representing words as continuous word vectors instead of one-hot representations and introducing a novel probabilistic loss to train these models as described in §5.1.1. First, we briefly summarize prior work that aimed at alleviating the softmax bottleneck,

highlighting conceptually different approaches.

## Softmax Alternatives

**Sampling-Based Approximations** Sampling-based approaches completely do away with computing the normalization term of softmax by considering only a small subset of possible outputs. These include approximations like Importance Sampling (Bengio and Senecal, 2003), Noise Constrastive Estimation (Mnih and Kavukcuoglu, 2013), Negative Sampling (Mikolov et al., 2013), and Blackout (Ji et al., 2016). These alternatives significantly speed-up training time but degrade generation quality.

**Structural Approximations** Morin and Bengio (2005) replace the flat softmax layer with a hierarchical layer in the form of a binary tree where words are at the leaves. This alleviates the problem of expensive normalization, but these gains are only obtained at training time. At test time, the hierarchical approximations lead to a drop in performance compared to softmax both in time efficiency and in accuracy. Chen et al. (2016) propose to divide the vocabulary into clusters based on their frequencies. Each word is produced by a different part of the hidden layer making the output embedding matrix much sparser. This leads to performance improvement both in training and decoding. However, it assigns fewer parameters to rare words which leads to inferior performance in predicting them (Ruder et al., 2019).

**Self Normalization Approaches** Andreas et al. (2015); Devlin et al. (2014) add additional terms to the training loss which makes the normalization factor close to 1, obviating the need to explicitly normalize. The evaluation of certain words can be done much faster than in softmax based models which is extremely useful for tasks like language modeling. However, for generation tasks, it is necessary to ensure that the normalization factor is exactly 1 which might not always be the case, and thus it might require explicit normalization.

**Character and Subword-Based Methods** Józefowicz et al. (2016) introduced character-based methods to reduce vocabulary size. Several studies (Al-Rfou et al., 2019; Choe et al., 2019; Xue et al., 2022) have since shown that character or byte-level models, if sufficiently large (more than 64 layers), can outperform other tokenization approaches. A major factor limiting their adoption is the fact that character sequences tend to be much longer, making training and inference slower. Sennrich et al. (2016) find a middle ground between characters and words based on sub-word units obtained using Byte Pair Encoding (BPE). BPE and its many variants have been shown to achieve good performance while also making the model truly open vocabulary. Since it is the state-of-the art approach currently used in most language generation models, we use it as a baseline in our experiments.

### 5.1.1 Methodology

In our proposed model, each word type in the output vocabulary is represented by a continuous vector $\mathbf{e}(w) \in \mathbb{R}^m$ where $m \ll V$. This representation can be obtained by training a word embedding model on a large monolingual corpus (Mikolov et al., 2013; Pennington et al., 2014; Bojanowski et al., 2017a).

At each generation step, the decoder of our model produces a continuous vector $\hat{\mathbf{e}} \in \mathbb{R}^m$. The output word is then predicted by searching for the nearest neighbor of $\hat{\mathbf{e}}$ in the embedding space:

$$w_{\text{predicted}} = \underset{w}{\operatorname{argmin}}\{d(\hat{\mathbf{e}}, \mathbf{e}(w)) | w \in \mathcal{V}\}$$

where $\mathcal{V}$ is the output vocabulary, $d$ is a distance function. In other words, the embedding space could be considered to be quantized into $V$ components and the generated continuous vector is mapped to a word based on the quanta in which it lies. The mapped word is then passed to the next step of the decoder (Gray, 1990). While training this model, we know the target vector $\mathbf{e}(w)$, and minimize its distance from the output vector $\hat{\mathbf{e}}$. With this formulation, our model is directly trained to optimize towards the information encoded by the embeddings. For example, if the embeddings are primarily semantic, as in Mikolov et al. (2013) or Bojanowski et al. (2017a), the model would tend to output words in a semantic space, that is produced words would either be correct or close synonyms (which we see in our analysis in §5.1.4), or if we use synactico-semantic embeddings (Levy and Goldberg, 2014; Ling et al., 2015), we might be able to also control for syntatic forms.

We propose a novel probabilistic loss function—a probabilistic variant of cosine loss—which gives a theoretically grounded regression loss for sequence generation and addresses the limitations of existing empirical losses (described in §5.1.2). Cosine loss measures the closeness between vector directions. A natural choice for estimating *directional* distributions is **von Mises-Fisher** (vMF) defined over a hypersphere of unit norm. That is, a vector close to the mean direction will have high probability. vMF is considered the directional equivalent of Gaussian distribution [2]. Given a target word $w$, its density function is given as follows:

$$p(\mathbf{e}(w); \boldsymbol{\mu}, \kappa) = C_m(\kappa) e^{\kappa \boldsymbol{\mu}^T \mathbf{e}(w)},$$

where $\boldsymbol{\mu}$ and $\mathbf{e}(w)$ are vectors of dimension $m$ with unit norm, $\kappa$ is a positive scalar, also called the concentration parameter. $\kappa = 0$ defines a uniform distribution over the hypersphere and $\kappa = \infty$ defines a point distribution at $\boldsymbol{\mu}$. $C_m(\kappa)$ is the normalization term:

$$C_m(\kappa) = \frac{\kappa^{m/2-1}}{(2\pi)^{m/2} I_{m/2-1}(\kappa)},$$

where $I_v$ is called modified Bessel function of the first kind of order $v$. The output of the model at each step is a vector $\hat{\mathbf{e}}$ of dimension $m$. We use $\kappa = \|\hat{\mathbf{e}}\|$. Thus the density function becomes:

$$p(\mathbf{e}(w); \hat{\mathbf{e}}) = \text{vMF}(\mathbf{e}(w); \hat{\mathbf{e}}) = C_m(\|\hat{\mathbf{e}}\|) e^{\hat{\mathbf{e}}^T \mathbf{e}(w)} \tag{5.2}$$

It is noteworthy that (5.2) is very similar to softmax computation (except that $\mathbf{e}(\mathbf{w})$ is a unit vector), the main difference being that normalization is not done by summing over the vocabulary, which makes it much faster than the softmax computation. More details about its computation are given in the appendix.

The negative log-likelihood of the vMF distribution, which at each output step is given by:

$$\text{NLLvMF}(\hat{\mathbf{e}}; \mathbf{e}(w)) = -\log\left(C_m(\|\hat{\mathbf{e}}\|)\right) - \hat{\mathbf{e}}^T \mathbf{e}(w)$$

**Regularization of NLLvMF** In practice, we observe that the NLLvMF loss puts too much weight on increasing $\|\hat{\mathbf{e}}\|$, making the second term in the loss function decrease rapidly without significant decrease in the cosine distance. To account for this, we add a regularization term. We experiment with two variants of regularization.

$\text{NLLvMF}_{\text{reg1}}$**:** We add $\lambda_1 \|\hat{\mathbf{e}}\|$ to the loss function, where $\lambda_1$ is a scalar hyperparameter.[3] This makes intuitive sense in that the length of the output vector should not increase too much. The regularized loss

[2] A natural choice for many regression tasks would be to use a loss function based on Gaussian distribution itself which is a probabilistic version of $\ell_2$ loss. But as we describe in §5.1.2, $\ell_2$ is not considered a suitable loss for regression on embedding spaces

[3] We empirically set $\lambda_1 = 0.02$ in all our experiments

function is as follows:

$$\text{NLLvMF}_{\text{reg}_1}(\hat{\mathbf{e}}) = -\log C_m(\|\hat{\mathbf{e}}\|) - \hat{\mathbf{e}}^T\mathbf{e}(w) + \lambda_1\|\hat{\mathbf{e}}\|$$

$\text{NLLvMF}_{\text{reg}_2}$**:** We modify the previous loss function as follows:

$$\text{NLLvMF}_{\text{reg}_2}(\hat{\mathbf{e}}) = -\log C_m(\|\hat{\mathbf{e}}\|) - \lambda_2\hat{\mathbf{e}}^T\mathbf{e}(w) \tag{5.3}$$

$-\log C_m(\|\hat{\mathbf{e}}\|)$ decreases slowly as $\|\hat{\mathbf{e}}\|$ increases as compared the second term. Adding a $\lambda_2 < 1$ the second term controls for how fast it can decrease.[4]

**Gradient Computation** The normalization constant $C_m(\kappa)$ is not directly differentiable because the Bessel function cannot be written in a closed form. The gradient of the first component ($\log(C_m\|\hat{\mathbf{e}}\|)$) of the loss is given as

$$\Delta \log(C_m(\kappa)) = -\frac{I_{m/2}(\kappa)}{I_{m/2-1}(\kappa)}.$$

This involves two computations of Bessel function ($I_v(z)$) for $m = 300$.[5]. For high values of $v$ and low values of $z$, the values of the Bessel function can become really small and lead to underflow (the gradient still being large). To deal with underflow, the gradient value can be approximated with its (tight) lower bound (Ruiz-Antolín and Segura, 2016)[6] ,

$$-\frac{I_{m/2}(\kappa)}{I_{m/2-1}(\kappa)} \geq -\frac{z}{v-1+\sqrt{(v+1)^2+z^2}}$$

That is, in the initial steps of training, one might need to use to the approximation of the gradient to train the model and switch to the actual computation later on. One could also approximate the value of $\log(C_m(\kappa))$ by integrating over the approximate gradient value which is given as

$$\log(C_m(\kappa)) \geq \sqrt{(v+1)^2+z^2} - (v-1)\log(v-1+\sqrt{(v+1)^2+z^2}).$$

In practice, we see that replacing $\log(C_m(\kappa))$ with this approximation in the loss function gives similar performance on the test data as well as alleviates the problem of underflow. We thus recommend using it.

### 5.1.2 Experiments: Machine Translation

#### Experimental Setup

We modify the standard seq2seq models in OpenNMT in PyTorch (Klein et al., 2017) for our experiments. The results presented in this chapter use a bidirectional LSTM encoder with an attention-based decoder (Luong et al., 2015).[7] The encoder has one layer whereas the decoder has 2 layers of size 1024 with the input word

[4] We use $\lambda_2 = 0.1$ in all our experiments

[5] we use `scipy.special.ive` for this purpose

[6] for $m = 300$, we don't face this issue, but it is useful if one is using embeddings of higher dimensions

[7] At the time of publishing this work, LSTM-based models were standard practice. We have since implemented and reproduced results on transformer-based (Vaswani et al., 2017a) models as well. See https://github.com/Sachin19/seq2seq-con for

embedding size of 512. For the baseline systems, the output at each decoder step multiplies a weight matrix ($HV$) followed by softmax. This model is trained until convergence on the validation perplexity. For our proposed models, we replace the softmax layer with the continuous output layer ($Hm$) where the outputs are $m$ dimensional. We empirically choose $m = 300$ for all our experiments. Additional hyperparameter and infrastructure details can be found in Table 5.1 and Table 5.2 respectively. These models are trained until convergence on the validation loss. Out-of-vocabulary words are mapped to an ⟨unk⟩ token[8]. We assign ⟨unk⟩ an embedding equal to the average of embeddings of all the words which are not present in the target vocabulary of the training set but are present in vocabulary on which the word embeddings are trained. Following Denkowski and Neubig (2017), after decoding a post-processing step replaces the ⟨unk⟩ token using a dictionary look-up of the word with the highest attention score. If the word does not exist in the dictionary, we back off to copying the source word itself. Bilingual dictionaries are automatically extracted from our parallel training corpus using word alignment (Dyer et al., 2013)[9]. We evaluate all the translations using BLEU score (Papineni et al., 2002).

We evaluate our systems on standard machine translation datasets from IWSLT'16 (Cettolo et al., 2016), on two target languages, English: German→English, French→English and a morphologically richer language French: English→French. The training sets for each of the language pairs contain around 220,000 parallel sentences. We use TED Test 2013+2014 (2,300 sentence pairs) as developments sets and TED Test 2015+2016 (2,200 sentence pairs) as test sets respectively for all the language pairs. All mentioned setups have a total vocabulary size of around 55,000 in the target language of which we choose top 50,000 words by frequency as the target vocabulary[10].

We also experiment with a much larger WMT'16 German→English (Bojar et al., 2016) task whose training set contains around 4.5M sentence pairs with the target vocabulary size of around 800,000. We use newstest2015 and newstest2016 as development and test data respectively. Since with continuous outputs we do not need to perform a time consuming softmax computation, we can train the proposed model with very large target vocabulary without any change in training time per batch. We perform this experiment with WMT'16 de–en dataset with a target vocabulary size of 300,000 (basically all the words in the target vocabulary for which we had trained embeddings). But to able to produce these words, the source vocabulary also needs to be increased to have their translations in the inputs, which would lead to a huge increase in the number of trainable parameters. Instead, we use sub-words computed using BPE as source vocabulary. We use 100,000 merge operations to compute the source vocabulary as we observe using a smaller number leads to too small (and less meaningful) sub-word units which are difficult to align with target words.

Both of these datasets contain examples from vastly different domains, while IWSLT'16 contains less formal spoken language, WMT'16 contains data primarily from news.

We train target word embeddings for English and French on corpora constructed using WMT'16 (Bojar et al., 2016) monolingual datasets containing data from Europarl, News Commentary, News Crawl from 2007 to 2015 and News Discussion (everything except Common Crawl due to its large memory requirements). These corpora consist of 4B+ tokens for English and 2B+ tokens for French. We experiment with two embedding models: word2vec (Mikolov et al., 2013) and fasttext (Bojanowski et al., 2017a) which were trained using the hyper-parameters recommended by the authors.

---

details.

[8] Although the proposed model can make decoding open vocabulary, there could still be unknown words, e.g., words for which we do not have pre-trained embeddings; we need ⟨unk⟩ token to represent these words

[9] https://github.com/clab/fast_align

[10] Removing the bottom 5,000 words did not make a significant difference in terms of translation quality

| Parameter | Value |
|---|---|
| LSTM Layers: Encoder | 1 |
| LSTM Layers: Decoder | 2 |
| Hidden Dimension (H) | 1024 |
| Input Word Embedding Size | 512 |
| Output Vector Size | 300 |
| Optimizer | Adam |
| Learning Rate (Baseline) | 0.0002 |
| Learning Rate (Our Models) | 0.0005 |
| Max Sentence Length | 100 |
| Vocabulary Size (Source) | 50000 |
| Vocabulary Size (Target) | 50000 |

Table 5.1: Hyperparameters Details

| | |
|---|---|
| PyTorch | 0.3.0 |
| CPU | Intel(R) Xeon(R) CPU 2.40GHz (32 Cores) |
| RAM | 190G |
| #GPUs/experiment | 1 |
| GPU | GeForce GTX TITAN X |

Table 5.2: Infrastructure details. All the experiments were run with this configuration

### Empirical Loss Functions

We compare our proposed loss function with standard loss functions used in multivariate regression.

**Squared Error** ($\ell_2$) is the most common distance function used when the model outputs are continuous (Lehmann and Casella, 1998). For each target word $w$, it is given as $\mathcal{L}_{\ell_2} = \|\hat{\mathbf{e}} - \mathbf{e}(w)\|^2$

$\ell_2$ penalizes large errors more strongly and therefore is sensitive to outliers. To avoid this we use a square rooted version of $\ell_2$ loss. But it has been argued that there is a mismatch between the objective function used to learn word representations (maximum likelihood based on inner product), the distance measure for word vectors (cosine similarity), and $\ell_2$ distance as the objective function to learn transformations of word vectors (Xing et al., 2015). This argument prompts us to look at cosine loss.

**Cosine Loss** is given as $\mathcal{L}_{\text{cosine}} = 1 - \frac{\hat{\mathbf{e}}^T \mathbf{e}(w)}{\|\hat{\mathbf{e}}\|.\|\mathbf{e}(w)\|}$. This loss minimizes the distance between the directions of output and target vectors while disregarding their magnitudes. The target embedding space in this case becomes a set of points on a hypersphere of dimension $m$ with unit radius.

**Max Margin Loss** Lazaridou et al. (2015) argue that using pairwise losses like $\ell_2$ or cosine distance for learning vectors in high dimensional spaces leads to *hubness*: word vectors of a subset of words appear as nearest neighbors of many points in the output vector space. To alleviate this, we experiment with a margin-based ranking loss (which has been shown to reduce hubness) to train the model to rank the word vector prediction $\hat{\mathbf{e}}$ for target vector $\mathbf{e}(w)$ higher than any other word vector $\mathbf{e}(w')$ in the embedding space. $\mathcal{L}_{\text{mm}} = \sum_{w' \in \mathcal{V}, w' \neq w} \max\{0, \gamma + \cos(\hat{\mathbf{e}}, \mathbf{e}(w')) - \cos(\hat{\mathbf{e}}, \mathbf{e}(w))\}$ where, $\gamma$ is a hyperparameter[11] representing the margin and $w'$ denotes negative examples. We use only one informative negative example as described in Lazaridou et al. (2015) which is closest to $\hat{\mathbf{e}}$ and farthest from the target word vector $\mathbf{e}(w)$. But, searching for this negative example requires iterating over the vocabulary which brings back the problem of slow loss computation.

### Decoding

In the case of empirical losses, we output the word whose target embedding is the nearest neighbor to the vector in terms of the distance (loss) defined. In the case of NLLvMF, we predict the word whose target embedding has the highest value of vMF probability density wrt to the output vector. This predicted word is fed as the input for the next time step. Our nearest-neighbor decoding scheme is equivalent to a greedy decoding; we thus compare to baseline models with beam size of 1.

[11] We use $\gamma = 0.5$ in our experiments.

| Embedding Model | Tied Emb | Source Type/ Target Type | Loss | BLEU | | |
|---|---|---|---|---|---|---|
| | | | | fr–en | de–en | en–fr |
| - | no | word→word | CE | 31.0 | 24.7 | 29.3 |
| - | no | word→BPE | CE | 29.1 | 24.1 | 29.8 |
| - | no | BPE→BPE | CE | **31.4** | **25.8** | **31.0** |
| word2vec | no | word→emb | L2 | 27.2 | 19.4 | 26.4 |
| word2vec | no | word→emb | Cosine | 29.1 | 21.9 | 26.6 |
| word2vec | no | word→emb | MaxMargin | 29.6 | 21.4 | 26.7 |
| fasttext | no | word→emb | MaxMargin | 31.0 | 25.0 | 29.0 |
| fasttext | yes | word→emb | MaxMargin | **32.1** | **25.0** | **31.0** |
| word2vec | no | word→emb | $\text{NLLvMF}_{\text{reg1}}$ | 29.5 | 22.7 | 26.6 |
| word2vec | no | word→emb | $\text{NLLvMF}_{\text{reg1+reg2}}$ | 29.7 | 21.6 | 26.7 |
| word2vec | yes | word→emb | $\text{NLLvMF}_{\text{reg1+reg2}}$ | 29.7 | 22.2 | 27.5 |
| fasttext | no | word→emb | $\text{NLLvMF}_{\text{reg1+reg2}}$ | 30.4 | 23.4 | 27.6 |
| fasttext | yes | word→emb | $\text{NLLvMF}_{\text{reg1+reg2}}$ | **32.1** | **25.1** | **31.7** |

Table 5.3: Translation quality experiments (BLEU scores) on IWSLT16 datasets

**Tying the target embeddings**

Until now we discussed the embeddings in the output layer. Additionally, decoder in a sequence-to-sequence model has an *input* embedding matrix as the previous output word is fed as an input to the decoder. Much of the size of the trainable parameters in all the models is occupied by these input embedding weights. We experiment with keeping this embedding layer fixed and tied with pre-trained target output embeddings (Press and Wolf, 2017). This leads to a significant reduction in the number of trainable parameters in our model.

### 5.1.3 Results

**Translation Quality** Table 5.3 shows the BLEU scores on the test sets for several baseline systems, and various configurations including the types of losses, types of inputs/outputs used (word, BPE, or embedding)[12] and whether the model used tied embeddings in the decoder or not.

$\ell_2$ loss attains the lowest BLEU scores among the proposed models; our manual error analysis reveals that the high error rate is due to the hubness phenomenon, as we described in §5.1.2. The BLEU scores improve for cosine loss, confirming the argument of Xing et al. (2015) that cosine distance is a better suited similarity (or distance) function for word embeddings. Best results—for MaxMargin and NLLvMF losses—surpass the strong BPE baseline in translation French→English and English→French, and attain slightly lower but competitive results on German→English.

Since we represent each target word by its embedding, the quality of embeddings should have an impact on the translation quality. We measure this by training our best model with fasttext embeddings (Bojanowski et al., 2017a), which leads to $> 1$ BLEU improvement. Tied embeddings are the most effective setups: they not only achieve highest translation quality, but also dramatically reduce parameters requirements and the speed of convergence.

Table 5.4 shows results on WMT'16 test set in terms of BLEU and METEOR (Denkowski and Lavie, 2014) trained only for best-performing setups in table 5.3. METEOR uses paraphrase tables and WordNet synonyms for common words. This may explain why METEOR scores, unlike BLEU, close the gap with the

[12]Note that we do not experiment with subword embeddings since the number of merge operations for BPE usually depend on the choice of a language pair which would require the embeddings to be retrained for every language pair.

baseline models: as we found in the qualitative analysis of outputs, our models often output synonyms of the reference words, which are plausible translations but are penalized by BLEU. [13] Examples are included in the Appendix.

| Loss | BLEU | METEOR |
|---|---|---|
| CE | 22.9 | 23.9 |
| CE (BPE) | 30.1 | 28.7 |
| MaxMargin | 24.3 | 25.2 |
| NLLvMF$_{reg_1+reg_2}$ | 28.8 | 28.2 |

Table 5.4: Translation quality experiment on WMT16 de–en

**Training Time** Table 5.6 shows the average training time per batch. In figure 5.1 (left), we show how many samples per second our proposed model can process at training time compared to the baseline. As we increase the batch size, the gap between the baseline and the proposed models increases. Our proposed models can process large mini-batches while still training much faster than the baseline models. The largest mini-batch size with which we can train our model is 512, compared to 184 in the baseline model. Using max-margin loss leads to a slight increase in the training time compared to NLLvMF. This is because its computation needs a negative example which requires iterating over the entire vocabulary. Since our model requires look-up of nearest neighbors in the target embedding table while testing, it currently takes similar time as that of softmax-based models. In future work, approximate nearest neighbors algorithms Johnson et al. (2017) can be used to improve translation time.

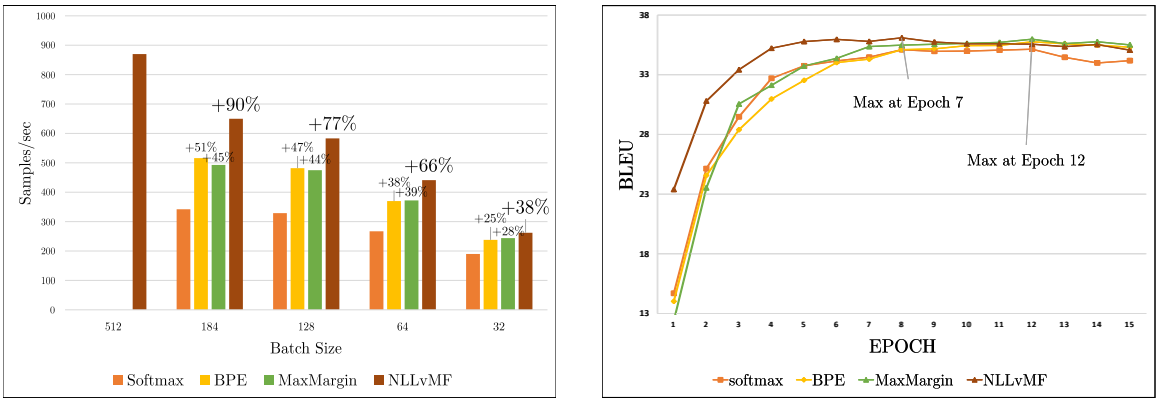



Figure 5.1: Left: Comparison of samples processed per second by the softmax vs. BPE vs. continuous output vMF models for IWSLT16 fr–en. Right: Comparison of convergence times of our models and baseline models on IWSLT16 fr–en validation sets. Baseline softmax as well as BPE converge at epoch 12 whereas our proposed model (NLLvMF) converges at epoch 7.

We also compare the speed of convergence, using BLEU scores on dev data. In figure 5.1 (right), we plot the BLEU scores against the number of epochs. Our model convergences much faster than the baseline models leading to an even larger improvement in overall training time (Similar figures for more datasets can be found in the appendix). As a result, as shown in table 5.5, the total training time of our proposed model (until convergence) is less than up-to 2.5x of the total training time of the baseline models.

[13] In IWSLT'16 datasets we obtain similar performances in BLEU and METEOR, this is likely because those models perform better particularly in translating rare words (§5.1.4) which are not covered in METEOR resources.

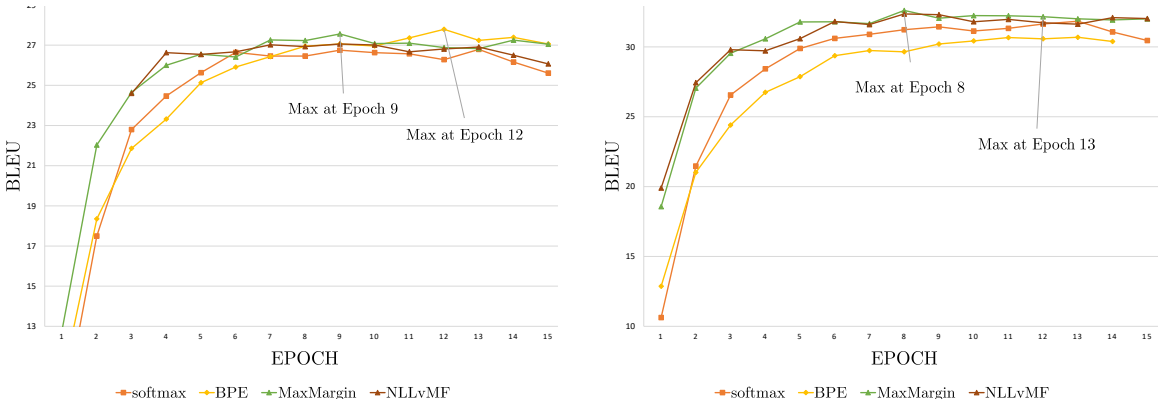



Figure 5.2: Comparison of convergence times of our models and baseline models on IWSLT16 de–en (left) and en–fr (right) validation sets.

| | **Softmax** | **BPE** | **Emb w/ NLL-vMF** |
|---|---|---|---|
| fr–en | 4h | 4.5h | **1.9h** |
| de–en | 3h | 3.5h | **1.5h** |
| en–fr | 1.8h | 2.8h | **1.3** |
| WMT de–en | 4.3d | 4.5d | **1.6d** |

Table 5.5: Total convergence times in hours(h)/days(d).

**Memory Requirements** As shown in Table 5.6 our best performing model requires less than 1% of the number of parameters in input and output layers, compared to BPE-based baselines.

| **Output Type** | **Tied** | **Loss** | **#Parameters Input Layer** | **#Parameters Output Layer** | **Training time** (ms) |
|---|---|---|---|---|---|
| word | No | CE | 25.6M (1.0x) | 51.2M (1.0x) | 400 (1.0x) |
| BPE | No | CE | 8.192M (0.32x) | 16.384M (0.32x) | 346 (0.86x) |
| emb | No | L2 | 25.6M (1.0x) | 307.2K (0.006x) | 160 (0.4x) |
| emb | No | Cosine | 25.6M (1.0x) | 307.2K (0.006x) | 160 (0.4x) |
| emb | No | MaxMargin | 25.6M (1.0x) | 307.2K (0.006x) | 178 (0.43x) |
| emb | Yes | MaxMargin | 153.6K (0.006x) | 307.2K (0.006x) | 178 (0.43x) |
| emb | No | NLLvMF$_x$ | 25.6M (1.0x) | 307.2K (0.006x) | 170 (0.42x) |
| emb | Yes | NLLvMF$_x$ | 153.6K (0.006x) | 307.2K (0.006x) | 170 (0.42x) |

Table 5.6: Comparison of number of parameters needed for input and output layer, train time per batch (with batch size of 64) for IWSLT16 fr–en. Numbers in parentheses indicate the fraction of parameters compared to word/word baseline model.

### 5.1.4 Analysis

**Translation of Rare Words** We evaluate the translation accuracy of words in the test set based on their frequency in the training corpus. Table 5.7 shows how the $F_1$ score varies with the word frequency. $F_1$ score gives a balance between recall (the fraction of words in the reference that the predicted sentence produces right) and precision (the fraction of produced words that are in reference). We show substantial improvements over softmax and BPE baselines in translating less frequent and rare words, which we hypothesize is due to having learned good embeddings of such words from the monolingual target corpus where these words

| Word Freq | Softmax | BPE | Max Margin | Emb w/ NLL-vMF |
|---|---|---|---|---|
| **1** | 0.42 | 0.50 | 0.30 | **0.52** |
| **2** | 0.16 | 0.26 | 0.25 | **0.31** |
| **3** | 0.14 | 0.22 | 0.25 | **0.33** |
| **4** | 0.29 | 0.24 | 0.30 | **0.33** |
| **5-9** | 0.28 | 0.33 | **0.38** | 0.37 |
| **10-99** | 0.54 | 0.53 | 0.53 | **0.55** |
| **100-999** | 0.60 | **0.61** | 0.60 | 0.60 |
| **1000+** | 0.69 | **0.70** | 0.69 | 0.69 |

Table 5.7: Test set unigram $F_1$ scores of occurrence in the predicted sentences based on their frequencies in the training corpus for different models for fr–en.

are not as rare. Moreover, in BPE based models, rare words on the source side are split in smaller units which are in some cases not properly translated in subword units on the target side if transparent alignments don't exist. For example, the word *saboter* in French is translated to *sab+ot+tate* by the BPE model whereas correctly translated as *sabotage* by our model. Also, a rare word *retraite* in French in translated to *pension* by both Softmax and BPE models (pension is a related word but less rare in the corpus) instead of the expected translation *retirement* which our model gets right.

We conducted a thorough analysis of outputs across our experimental setups. Few examples are shown in the next section. Interestingly, there are many examples where our models do not exactly match the reference translations (so they do not benefit from in terms of BLEU scores) but produce meaningful translations. This is likely because the model produces nearby words of the target words or paraphrases instead of the target word (which are many times synonyms).

Since we are predicting embeddings instead of actual words, the model tends to be weaker sometimes and does not follow a good language model and leads to ungrammatical outputs in cases where the baseline model would perform well. Integrating a pre-trained language model within the decoding framework is one potential avenue for our future work. Another reason for this type of errors could be our choice of target embeddings which are not modeled to (explicitly) capture syntactic relationships. Using syntactically inspired embeddings (Levy and Goldberg, 2014; Ling et al., 2015) might help reduce these errors. However, such fluency errors are not uncommon also in softmax and BPE-based models either.

**Beam Search** In Table 5.3, we present results of translation quality with our proposed model and comparable baselines with a beam size of one. Here, for completeness, table 5.8 shows additional results with softmax-based models with a beam size of 5.

| Loss | BLEU |
|---|---|
| IWSLT fr–en | 32.2 |
| IWSLT de–en | 26.1 |
| IWSLT en–fr | 32.4 |
| WMT de–en | 31.9 |

Table 5.8: Translation quality experiments using beam search with BPE based baseline models with a beam size of 5

With our proposed models, in principle, it is possible to generate candidates for beam search by using $K$-Nearest Neighbors. But how to rank the partially generated sequences is not trivial (one could use the loss

values themselves to rank, but initial experiments with this setting did not result in significant gains). In this work, we focus on enabling training with continuous outputs efficiently and accurately giving us huge gains in training time. The question of decoding with beam search requires substantial investigation and we leave it for future work.

### 5.1.5 Examples

Table 5.9 and Table 5.10 provide selected examples generated by our proposed approach and the baselines highlighting the kinds of errors made by both.

| | |
|---|---|
| Input | Une éducation est critique, mais régler ce problème va nécessiter que chacun d'entre nous s'engage et soit un meilleur exemple pour les femmes et filles dans nos vies. |
| Reference | An education is critical, but tackling this problem is going to require each and everyone of us to step up and be better role models for the women and girls in our own lives. |
| Predicted (BPE) | Education is critical, but it's going to require that each of us *will come in and if you do* a better **example** for women and girls in our lives. |
| Predicted (L2) | Education is critical , but to to do this is going to require that each of us *of* to engage and *or* a better **example** *of* the women and girls in our lives. |
| Predicted (Cosine) | That's critical , but *that's that it's* going to require that each of us is going to *take that* the problem and they're going *to if* you're a better **example** for women and girls in our lives. |
| Predicted (MaxMargin) | Education is critical, but that problem is going to require that every one of us **is engaging** and **is a better example** for women and girls in our lives. |
| Predicted ($\text{NLLvMF}_{reg}$) | Education is critical , but *fixed* this problem is going to require that **all of us engage** and **be a better example** for women and girls in our lives. |

Table 5.9: Translation examples. Red and blue colors highlight translation errors; red are bad and blue are outputs that are good translations, but are considered as errors by the BLEU metric. Our systems tend to generate a lot of such "meaningful" errors.

| | |
|---|---|
| Input | Pourquoi ne sommes nous pas de simples robots qui traitent toutes ces données, produisent ces résultats, sans faire l'expérience de ce film intérieur ? |
| Reference | Why aren't we just robots who process all this input, produce all that output, without experiencing the inner movie at all? |
| Predicted (BPE) | Why *don't we have* simple robots that *are processing* all of this data, produce **these results**, without *doing* **the experience of that** inner movie? |
| Predicted (L2) | Why are we not *that we do that that are technologized and that that that's all* **these results**, *that they're actually doing these results, without do* **the experience of this film inside**? |
| Predicted (Cosine) | Why are we not simple robots that *all that* data and produce these *data* **without the experience of this film inside**? |
| Predicted (MaxMargin) | Why aren't we just simple robots that have all this data, make these results, without *making* **the experience of this inside** movie? |
| Predicted ($\text{NLLvMF}_{reg}$) | Why are we not simple robots that treat all this data, produce **these results**, without **having the experience of this inside film**? |

Table 5.10: Example of fluency errors in the baseline model. Red and blue colors highlight translation errors; red are bad and blue are outputs that are good translations, but are considered as errors by the BLEU metric.

### 5.1.6 Extensions to this work

**Phrase-based NMT** Park and Tsvetkov (2019) extend this framework to pretrain word *and* phrase embeddings for salient phrases and show that it improves output quality when translating from morphologically rich languages like German and Turkish to English while improving the training speed substantially.

**Paraphrasing via Multilingual Models** Our qualitative analysis in Kumar and Tsvetkov (2019) reveals that predicting embeddings often leads to predicting synonyms of target words. We exploit this observation in Jegadeesan et al. (2021) and adapt this approach for bilingual translation using a shared model (e.g. French to English and English to French) and show that it enables generating paraphrases in both languages with much higher diversity that training the same model with a softmax-based loss.

## 5.2 Machine Translation Into Low-Resource Language Varieties

Despite tremendous progress in machine translation (Bahdanau et al., 2015b; Vaswani et al., 2017a) and language generation in general, current state-of-the-art systems typically only work under the assumption that a language is homogeneously spoken and understood by its speakers: they generate a "standard" form of the target language, typically based on the availability of parallel data. But language use varies with regions, socio-economic backgrounds, ethnicity, and fluency, and many widely spoken languages consist of dozens of varieties or dialects, with differing lexical, morphological, and syntactic patterns for which no translation data are typically available. As a result, models trained to translate from a source language (SRC) to a standard language variety (STD) lead to a sub-par experience for speakers of other varieties.

Motivated by these issues, in this work, we present approaches to adapt trained SRC→STD translation models to generate text in a different target variety (TGT), having access only to limited monolingual corpora in TGT and no SRC–TGT parallel data. TGT may be a dialect of, a language variety of, or a typologically-related language to STD.

We present an effective transfer-learning framework for translation into low-resource language varieties. Our method reuses SRC→STD MT models presented in §5.1.1 and finetunes them on synthesized (pseudo-parallel) SRC–TGT texts. This allows for a rapid adaptation of MT models to new varieties without having to train everything from scratch. Using word-embedding adaptation techniques, we show that MT models which predict continuous word vectors (Kumar and Tsvetkov, 2019) rather than softmax probabilities lead to superior performance since they allow additional knowledge to be injected into the models through transfer between word embeddings of high-resource (STD) and low-resource (TGT) monolingual corpora.

We evaluate our framework on three translation tasks: English to Ukrainian and Belarusian, assuming parallel data are only available for English→Russian; English to Nynorsk, with only English to Norwegian Bokmål parallel data; and English to four Arabic dialects, with only English→Modern Standard Arabic (MSA) parallel data. Our approach outperforms competitive baselines based on unsupervised MT, and methods based on finetuning softmax-based models.

### 5.2.1 A Transfer-learning Architecture

We first formalize the task setup. We are given a parallel SRC→STD corpus, which allows us to train a translation model $f(\cdot;\theta)$ that takes an input sentence $x$ in SRC and generates its translation in the standard veriety STD, $\hat{y}_{\text{STD}} = f(x;\theta)$. Here, $\theta$ are the learnable parameters of the model. We are also given monolingual

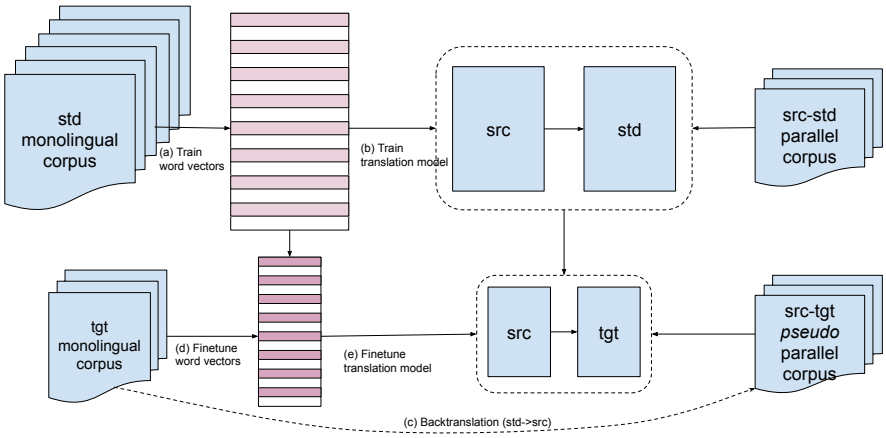



Figure 5.3: An overview of our approach. (a) Using the available STD monolingual corpora, we first train word vectors using `fasttext`; (b) we then train a SRC→STD translation model using the parallel corpora to predict the pretrained word vectors; (c) next, we train STD→SRC model and use it to translate TGT monolingual corpora to SRC; (d) now, we finetune STD subword embeddings to learn TGT word embeddings; and finally (e) we finetune a SRC→STD model to generate TGT pretrained embeddings using the back-translated SRC→TGT data.

corpora in both the standard STD and target variety TGT. Our goal now is to modify $f$ to generate translations $\hat{y}_{\text{TGT}}$ in the target variety TGT. At training time, we assume no SRC–TGT or STD–TGT parallel data are available.

Our solution (Figure 5.3) is based on a transformer-based encoder-decoder architecture (Vaswani et al., 2017a) which we modify to predict word vectors. Instead of treating each token in the vocabulary as a discrete unit, we represent it using a unit-normalized $d$-dimensional pre-trained vector. These vectors are learned from a STD monolingual corpus using `fasttext` (Bojanowski et al., 2017b). A word's representation is computed as the average of the vectors of its character $n$-grams, allowing surface-level linguistic information to be shared among words. At each step in the decoder, we feed this pretrained vector at the input and instead of predicting a probability distribution over the vocabulary using a softmax layer, we predict a $d$-dimensional continuous-valued vector. We train this model by minimizing the von Mises-Fisher (vMF) loss—a probabilistic variant of cosine distance—between the predicted vector and the pre-trained vector. The pre-trained vectors (at both input and output of the decoder) are not trained with the model. To decode from this model, at each step, the output word is generated by finding the closest neighbor (in terms of cosine similarity) of the predicted output vector in the pre-trained embedding table.

We train $f$ in this fashion using SRC–STD parallel data. As shown below, training a softmax-based SRC→STD model to later finetune with TGT suffers from vocabulary mismatch between STD and TGT and thus is detrimental to downstream performance. By replacing the decoder input and output with pretrained vectors, we separate the vocabulary from the MT model, making adaptation easier.

Now, to finetune this model to generate TGT, we need TGT embeddings. Since the TGT monolingual corpus is small, training `fasttext` vectors on this corpus from scratch will lead (as we show) to low-quality embeddings. Leveraging the relatedness of STD and TGT and their vocabulary overlap, we use STD embeddings to transfer knowledge to TGT embeddings: for each character $n$-gram in the TGT corpus, we initialize its embedding with the corresponding STD embedding, if available. We then continue training `fasttext` on the TGT monolingual corpus (Chaudhary et al., 2018). Last, we use a supervised embedding alignment method (Lample et al., 2018a) to project the learned TGT embeddings in the same space as STD. STD and TGT are expected to have a large lexical overlap, so we use identical tokens in both varieties as supervision for this alignment. The obtained embeddings, due to transfer learning from STD, inject additional knowledge in the

model.

Finally, to obtain a SRC→TGT model, we finetune $f$ on psuedo-parallel SRC–TGT data. Using a STD→SRC MT model (a back-translation model trained using large STD–SRC parallel data with standard settings) we (back)-translate TGT data to SRC. Naturally, these synthetic parallel data will be noisy despite the similarity between STD and TGT, but we show that they improve the overall performance. We discuss the implications of this noise in §5.2.3.

### 5.2.2 Experimental Setup

**Datasets** We experiment with two setups. In the first (synthetic) setup, we use English (EN) as SRC, Russian (RU) as STD, and Ukrainian (UK) and Belarusian (BE) as TGTs. We sample 10M EN-RU sentences from the WMT'19 shared task (Ma et al., 2019), and 80M RU sentences from the CoNLL'17 shared task to train embeddings. To simulate low-resource scenarios, we sample 10K, 100K and 1M UK sentences from the CoNLL'17 shared task and BE sentences from the OSCAR corpus (Ortiz Suárez et al., 2020). We use TED dev/test sets for both languages pairs (Cettolo et al., 2012).

The second (real world) setup has two language sets: the first one defines English as SRC, with Modern Standard Arabic (MSA) as STD and four Arabic varieties spoken in Doha, Beirut, Rabat and Tunis as TGTs. We sample 10M EN-MSA sentences from the UNPC corpus (Ziemski et al., 2016), and 80M MSA sentences from the CoNLL'17 shared task. For Arabic varieties, we use the MADAR corpus (Bouamor et al., 2018) which consists of 12K 6-way parallel sentences between English, MSA and the 4 considered varieties. We ignore the English sentences, sample dev/test sets of 1K sentences each, and consider 10K monolingual sentences for each TGT variety. The second set also has English as SRC with Norwegian Bokmål (NO) as STD and its written variety Nynorsk (NN) as TGT. We use 630K EN-NO sentences from WikiMatrix (Schwenk et al., 2021), and 26M NO sentences from ParaCrawl (Esplà et al., 2019) combined with the WikiMatrix NO sentences to train embeddings. We use 310K NN sentences from WikiMatrix, and TED dev/test sets for both varieties (Reimers and Gurevych, 2020a).

**Preprocessing** We preprocess raw text using Byte Pair Encoding (BPE, Sennrich et al., 2016) with 24K merge operations on each SRC–STD corpus trained separately on SRC and STD. We use the same BPE model to tokenize the monolingual STD data and learn `fasttext` embeddings (we consider character $n$-grams of length 3 to 6).[14] Splitting the TGT words with the same STD BPE model will result in heavy segmentation, especially when TGT contains characters not present in STD.[15] To counter this, we train a joint BPE model with 24K operations on the concatenation of STD and TGT corpora to tokenize TGT corpus following Chronopoulou et al. (2020). This technique increases the number of shared tokens between STD and TGT, thus enabling better cross-variety transfer while learning embeddings *and* while finetuning. We follow Chaudhary et al. (2018) to train embeddings on the generated TGT vocabulary where we initialize the character $n$-gram representations for TGT words with STD's `fasttext` model wherever available and finetune them on the TGT corpus.

**Implementation and Evaluation** We modify the standard OpenNMT-py seq2seq models of PyTorch (Klein et al., 2017) to train our model with vMF loss (Kumar and Tsvetkov, 2019). We use the transformer-BASE model (Vaswani et al., 2017a), with 6 layers in both encoder and decoder and with 8 attention heads, as our

[14] We slightly modify `fasttext` to not consider BPE token markers "@@" in the character $n$-grams.

[15] For example, both RU and UK alphabets consist of 33 letters; RU has the letters Ёё, ъ, ы and Ээ, which are not used in UK. Instead, UK has Ґґ, Єє, Іі and Її.

| Size of TGT corpus | UK 10K | UK 100K | UK 1M | BE 10K | BE 100K | BE 1M | NN 300K | Arabic Varieties (10K) Doha | Beirut | Rabat | Tunis |
|---|---|---|---|---|---|---|---|---|---|---|---|
| SUP(SRC→STD) | 1.7 | 1.7 | 1.7 | 1.5 | 1.5 | 1.5 | 11.3 | 3.7 | 1.8 | 2.0 | 1.3 |
| UNSUP(SRC→TGT) | 0.3 | 0.6 | 0.9 | 0.4 | 0.6 | 1.4 | 2.7 | 0.2 | 0.1 | 0.1 | 0.1 |
| PIVOT | 1.5 | 8.6 | 14.9 | 1.15 | 3.9 | 8.0 | 11.9 | 1.8 | 2.1 | 1.7 | 1.1 |
| SOFTMAX | 1.9 | 12.7 | 15.4 | 1.5 | 4.5 | 7.9 | 14.4 | 14.5 | 7.4 | 4.9 | 3.9 |
| **LANGVARMT** | **6.1** | **13.5** | **15.3** | **2.3** | **8.8** | **9.8** | **16.6** | **20.1** | **8.1** | **7.4** | **4.6** |

Table 5.11: BLEU scores on translation from English to Ukrainian, Belarusian, Nynorsk, and Arabic dialects with varying amounts of monolingual target data (TGT sentences) available for finetuning. Our approach (LANGVARMT) outperforms all baselines.

underlying architecture. We modify this model to predict pretrained `fasttext` vectors. We also initialize the decoder input embedding table with the pretrained vectors and do not update them during model training. All models are optimized using Rectified Adam (Liu et al., 2020) with a batch size of 4K tokens and dropout of 0.1. We train SRC→STD models for 350K steps with an initial learning rate of 0.0007 with linear decay. For finetuning, we reduce the learning rate to 0.0001 and train for up to 100K steps. We use early stopping in all models based on validation loss computed every 2K steps. We decode all the softmax-based models with a beam size of 5 and all the vMF-based models greedily.

We evaluate our methods using BLEU score (Papineni et al., 2002) based on the SacreBLEU implementation (Post, 2018). While we recognize the limitations of BLEU (Mathur et al., 2020), more sophisticated embedding-based metrics for MT evaluation (Zhang et al., 2020; Sellam et al., 2020) are simply not available for language varieties. For the Arabic varieties, we also report a macro-average. In addition, to measure the expected impact on actual systems' users, we follow Faisal et al. (2021) in computing a population-weighted macro-average ($\textbf{avg}_{\textbf{pop}}$) based on language community populations provided by Ethnologue Eberhard et al. (2019).

**Baselines** Our proposed framework, **LANGVARMT**, consists of three main components: (1) A supervised SRC→STD model is trained to predict continuous STD word embeddings rather than discrete softmax probabilities. (2) Output STD embeddings are replaced with TGT embeddings. The TGT embeddings are trained by finetuning STD embeddings on monolingual TGT data and aligning the two embedding spaces. (3) The resulting model is finetuned with pseudo-parallel SRC→TGT data.

We compare LANGVARMT with the following competitive baselines. **SUP(SRC→STD)**: train a standard (softmax-based) supervised SRC→STD model, and consider the output of this model as TGT under the assumption that STD and TGT may be very similar. **UNSUP(SRC→TGT)**: train an unsupervised MT model (Lample et al., 2018a) in which the encoder and decoder are initialized with cross-lingual masked language models (MLM, Conneau and Lample, 2019). These MLMs are pre-trained on SRC monolingual data, and then finetuned on TGT monolingual data with an expanded vocabulary as described above. This baseline is taken from Chronopoulou et al. (2020), where it showed state-of-the-art performance for low-monolingual-resource scenarios. **Pivot**: train a UNSUP(STD→TGT) model as described above using STD and TGT monolingual corpora. During inference, translate the SRC sentence to STD with the SUP(SRC→STD) model and then to TGT with the UNSUP(STD→TGT) model. We also perform several ablation experiments, showing that every component of LANGVARMT is necessary for good downstream performance. Specifically, we report results with LANGVARMT but using a standard softmax layer (**SOFTMAX**) to predict tokens instead of continuous vectors.

### 5.2.3 Results and Analysis

Table 5.11 compares the performance of LANGVARMT with the baselines for Ukrainian, Belarusian, Nynorsk, and the four Arabic varieties. For reference, note that the EN→RU, EN→MSA, and EN→NO models are relatively strong, yielding BLEU scores of 24.3, 21.2, and 24.9, respectively.

**Synthetic Setup** Considering STD and TGT as the same language is sub-optimal, as is evident from the poor performance of the non-adapted SUP(SRC→STD) model. Clearly, special attention ought to be paid to language varieties. Direct unsupervised translation from SRC to TGT performs poorly as well, confirming previously reported results of the ineffectiveness of such methods on unrelated languages Guzmán et al. (2019).

Translating SRC to TGT by pivoting through STD achieves much better performance owing to strong UNSUP(STD→TGT) models that leverage the similarities between STD and TGT. However, when resources are scarse (e.g., with 10K monolingual sentences as opposed to 1M), this performance gain considerably diminishes. We attribute this drop to overfitting during the pre-training phase on the small TGT monolingual data. Ablation results also show that in such low-resource settings the learned embeddings are of low quality.

Finally, LANGVARMT consistently outperforms all baselines. Using 1M UK sentences, it achieves similar performance (for EN→UK) to the softmax ablation of our method, SOFTMAX, and small gains over unsupervised methods. However, in lower resource settings our approach is clearly better than the strongest baselines by over 4 BLEU points for UK (10K) and 3.9 points for BE (100K). On our resource-richest setup of EN→UK translation using 1M UK sentences and RU as STD, we compare our method with the following additional baselines.

| Method | BLEU (uk) |
|---|---|
| SUP(SRC-STD) | 1.7 |
| UNSUP(SRC→TGT) | 0.9 |
| PIVOT: | 14.9 |
| LAMPLE-UNSUP(SRC→TGT) | 0.4 |
| PIVOT:LAMPLE-UNSUP(STD→TGT) | 9.0 |
| PIVOT:DICTREPLACE(STD→TGT) | 2.9 |
| LANGVARMT | 15.3 |
| LANGVARMT w/ poor embeddings | 4.6 |
| LANGVARMT-RANDOM | 13.1 |
| SOFTMAX | 15.4 |
| LANGVARMT-RANDOM-SOFTMAX | 14.1 |

Table 5.12: BLEU scores on EN-UK test corpus with 1M UK monolingual corpus.

LAMPLE-UNSUP(SRC→TGT): This is another unsupervised model, based on Lample et al. (2018a) which initializes the input and output embedding tables of both encoder and decoder using cross-lingual word embeddings trained on SRC and TGT monolingual corpora. The model is trained in a similar manner to Chronopoulou et al. (2020) (UNSUP(SRC→TGT)) with iterative backtranslation and autoencoding.

PIVOT:LAMPLE(STD→TGT): This baseline is similar to the PIVOT baseline, where we replace the unsupervised model with that of Lample et al. (2018a).

PIVOT:DICTREPLACE(STD→TGT): Here we first translate SRC to STD using SUP(SRC→STD), and then modify the STD output to get a TGT sentence as follows: We create a STD–TGT dictionary using the embedding map suggested by Lample et al. (2018b). This dictionary is created on words tokenized with Moses tokenizer (Hoang and Koehn, 2008) rather than BPE tokens. We replace each token in the generated STD

sentence which is not in the TGT vocabulary using the dictionary (if available). We consider this baseline to measure lexical vs. syntactic/phrase level differences between Russian and Ukrainian.

In addition to baseline comparison, we report the following ablation experiments. (1) To measure transfer from STD to TGT embeddings, we finetune the SUP(SRC→STD) model using TGT embeddings trained from scratch (as opposed to initialized with STD embeddings). (2) To measure the impact of initialization during model finetuning, we compare with a randomly initialized model trained in a supervised fashion on the psuedo-parallel SRC–TGT data.

On the unsupervised models based on Lample et al. (2018a), we observe a similar trend as that of Chronopoulou et al. (2020), where the LAMPLE-UNSUP(SRC→TGT) model performing poorly (0.4) with substantial gains when pivoting through Russian (9.0 BLEU). PIVOT:DICTREPLACE(STD→TGT) gains some improvement over considering the output of SUP(SRC→STD) as TGT, probably due to syntactic similarities between Russian and Ukrainian. This result can potentially be further improved with a human-curated RU–UK dictionary, but such resources are typically not available for the low-resource settings we consider in this paper.

As shown in Table 5.12, training the SRC→TGT model on a randomly initialized model (LANGVAR-RANDOM) results in a performance drop, confirming that transfer learning from a SRC→STD model is beneficial. Similarly, using TGT embeddings trained from scratch (LANGVARMT w/ poor embeddings) results in a drastic performance drop, providing evidence for essential transfer from STD embeddings.

**Real-world Setup** The effectiveness of LANGVARMT is pronounced in this setup with a dramatic improvement of more than 18 BLEU points over unsupervised baselines when translating into Doha Arabic. We hypothesize that during the pretraining phase of unsupervised methods, the extreme difference between the size of the MSA monolingual corpus (10M) and the varieties' corpora (10K) leads to overfitting. Additionally, compared to the synthetic setup, the Arabic varieties we consider are quite close to MSA, allowing for easy and effective adaptation of both word embeddings and EN→MSA models. LANGVARMT also improves in all other Arabic varieties, although naturally some varieties remain challenging. For example, the Rabat and particularly the Tunis varieties are more likely to include French loanwords Bouamor et al. (2018) which are not adequately handled as they are not part of our vocabulary. In future work, we will investigate whether we can alleviate this issue by potentially including French corpora (transliterated into Arabic) to our TGT language corpora. On average, our approach improves by 2.3 BLEU points over the softmax-based baseline (cf. 7.7 and 10.0 in Table 5.13 under $\textbf{avg}_{\mathcal{L}}$) across the four Arabic dialects. For a population-weighted average ($\textbf{avg}_{\textbf{pop}}$), we associate the Doha variety with Gulf Arabic (ISO code: `afb`), the Beirut one with North Levantine Arabic (`apc`), Rabat with Moroccan (`ary`), and the Tunis variety with Tunisian Arabic (`aeb`). As before, LANGVARMT outperforms the baselines. The absolute BLEU scores in this highly challenging setup are admittedly low, but as we discuss in ablations above, the translations generated by LANGVARMT are often fluent and input-preserving, especially compared to the baselines.

Finally, due to the high similarity between NO and NN, the SUP(EN→NO) model also performs well on NN with 11.3 BLEU, but our method yields further gains of over 4 points over the baselines.

### 5.2.4 Discussion and Analysis

To better understand the performance of our models, we perform the following additional analyses.

**Fairness** The goal of this work is to develop more equitable technologies, usable by speakers of diverse language varieties. When evaluating multilingual and multi-dialect systems, it is crucial that the evaluation

takes into account principles of fairness, as outlined in economics and social choice theory Choudhury and Deshpande (2021). We follow the least difference principle proposed by Rawls (1999), whose egalitarian approach proposes to narrow the gap between unequal accuracies.

A simple proxy for unfairness is the standard deviation (or, even simpler, a $\max - \min$ performance) of the scores across languages. Beyond that, we measure a system's *unfairness* with respect to the different subgroups using the adaptation of generalized entropy index described by Speicher et al. (2018), which considers equities within and between subgroups in evaluating the overall unfairness of an algorithm on a population. The generalized entropy index for a population of $n$ individuals receiving benefits $b_1, b_2, \ldots, b_n$ with mean benefit $\mu$ is

$$\mathcal{E}^{\alpha}(b_1, \ldots, b_n) = \frac{1}{n\alpha(\alpha-1)} \sum_{i=1}^{n} \left[ \left( \frac{b_i}{\mu} \right)^{\alpha} - 1 \right].$$

Using $\alpha = 2$ following Speicher et al. (2018), the generalized entropy index corresponds to half the squared coefficient of variation.[16]

If the underlying population can be split into $|G|$ disjoint subgroups across some attribute (e.g. gender, age, or language variety) we can decompose the total unfairness into individual and group-level unfairness. Each subgroup $g \in G$ will correspond to $n_g$ individuals with corresponding benefit vector $\mathbf{b}^g = (b_1^g, b_2^g, \ldots, b_{n_g}^g)$ and mean benefit $\mu_g$. Then, total generalized entropy can be re-written as:

$$\begin{aligned}
\mathcal{E}^{\alpha}(b_1, \ldots, b_n) =& \sum_{g=1}^{|G|} \frac{n_g}{n} \left( \frac{\mu_g}{\mu} \right)^{\alpha} \mathcal{E}^{\alpha}(\mathbf{b}^g) \\
&+ \sum_{g=1}^{|G|} \frac{n_g}{n\alpha(\alpha-1)} \left[ \left( \frac{\mu_g}{\mu} \right)^{\alpha} - 1 \right] \\
=& \mathcal{E}^{\alpha}(\mathbf{b}) + \mathcal{E}_{\beta}^{\alpha}(\mathbf{b}).
\end{aligned}$$

The first term $\mathcal{E}^{\alpha}(\mathbf{b})$ corresponds to the weighted unfairness score that is observed *within* each subgroup, while the second term $\mathcal{E}_{\beta}^{\alpha}(\mathbf{b})$ corresponds to the unfairness score *across* different subgroups.

In this measure of unfairness, we define the benefit as being directly proportional to the system's accuracy. For a Machine Translation system, each user receives an average benefit equal to the BLEU score the MT system achieves on the user's dialect. Conceptually, if the system produces a perfect translation (BLEU=1) then the user will receive the highest benefit of 1. If the system fails to produce a meaningful translation (BLEU$\rightarrow$ 0) then the user receives no benefit ($b = 0$) from the interaction with the system.

Table 5.13 reports different Arabic multi-dialect systems' unfairness. We find that our proposed method is fairer across all dialects, compared to baselines where only MSA translation produces comprehensible outputs.

**Lemmatized BLEU** To identify potential sources of error in our proposed method, for UK and BE, we lemmatize the generated translations and the references and re-compute BLEU scores (Qi et al., 2020). The results are summarized in Table 5.14. Across all data sizes, both UK and BE achieve a substantial increase in BLEU (up to +6 BLEU) compared to that obtained on raw text, likely indicating that our framework often generates correct lemmas, but may fail on the correct inflectional form of the target words. This highlights the importance of considering morphological differences between language varieties. The high BLEU scores

[16] The coefficient of variation is simply the ratio of the standard deviation $\sigma$ to the mean $\mu$ of a distribution.

| Model | $\textbf{avg}_{\mathcal{L}}\uparrow$ | $\textbf{avg}_{\textbf{pop}}\uparrow$ | max−min↓ | unfair↓ |
|---|---|---|---|---|
| SUP(SRC→STD) | 2.2 | 1.8 | 19.9 | 0.037 |
| UNSUP(SRC→TGT) | 0.1 | 0.1 | 21.1 | 0.046 |
| PIVOT | 1.7 | 1.8 | 20.1 | 0.037 |
| SOFTMAX | 7.7 | 5.7 | 17.3 | 0.020 |
| **LANGVARMT** | **10.0** | **7.3** | **16.6** | **0.016** |

Table 5.13: Average performance and fairness metrics across the four Arabic varieties. This evaluation includes MSA (with a BLEU score of 21.2 on the SUP(EN→MSA) model).

also demonstrate that the resulting translations are quite likely understandable, albeit not always grammatical. Future work may investigate alleviating these issues by considering TGT embeddings based on morphological features of tokens (Chaudhary et al., 2018).

| | EN→UK | | | EN→BE | | |
|---|---|---|---|---|---|---|
| | 10K | 100K | 1M | 10K | 100K | 1M |
| raw | 6.1 | 13.5 | 15.3 | 2.3 | 8.8 | 9.8 |
| lemma | 12.8 | 19.5 | 21.3 | 3.5 | 13.7 | 15.8 |

Table 5.14: BLEU scores on raw vs lemmatized text with LANGVARMT.

**Translation of Rare Words** On the outputs of the EN→UK model, trained with 100K UK sentences, we compute the translation accuracy of words based on their frequency in the TGT monolingual corpus for LANGVARMT, our best baseline SUP(SRC→STD)+UNSUP(SRC→TGT) and the best performing ablation SOFTMAX. These results, shown in Table 5.15, reveal that LANGVARMT is more accurate at translating rare words (with frequency less than 10) compared to the baselines.

### Examples

We provide some examples of EN-UK and EN-Beirut Arabic translations generated by the three models in Tables 5.16 and 5.17. As evaluated by native speakers of the Beirut Arabic, we find that despite a BLEU score of only 8, in a majority of cases our baseline model is able to generate fluent translations of the input, preserving most of the content, whereas the baseline model ignores many of the content words. We also observe that in some cases, despite predicting in the right semantic space of the pretrained embeddings, it fails to predict the right token, resulting in surface form errors (e.g., predicting adjectival forms of verbs).

### Negative Results

We present results for the following experiments: (a) adapting an English to Thai (EN→TH) model to Lao (LO). We use a parallel corpus of around 10M sentences for training the supervised EN→TH model from the CCAligned corpus (El-Kishky et al., 2020), around 140K LO monolingual sentences from the OSCAR corpus (Ortiz Suárez et al., 2020) and TED2020 dev/tests for both TH and LO[17] (Reimers and Gurevych, 2020a). (b) adapting an English to Amharic Model (EN→AM) to Tigrinya (TI). We use training, development and test sets from the JW300 corpus (Agić and Vulić, 2019) containing 500K EN–AM parallel corpus and 100K Tigrinya monolingual sentences.

[17]Although Thai and Lao scripts look very similar, they use different Unicode symbols which are one-to-one mappable to each other: https://en.wikipedia.org/wiki/Lao_(Unicode_block)

| frequency | PIVOT | SOFTMAX | LANGVARMT |
|---|---|---|---|
| 1 | 0.0429 | 0.1516 | 0.1812 |
| 2 | 0.0448 | 0.2292 | 0.2556 |
| 3 | 0.0597 | 0.2246 | 0.2076 |
| 4 | 0.0692 | 0.2601 | 0.2962 |
| [5,10) | 0.0582 | 0.2457 | 0.2722 |
| [10,100) | 0.1194 | 0.2881 | 0.2827 |
| [100,1000) | 0.2712 | 0.4537 | 0.4449 |

Table 5.15: Translation accuracies of words based on their frequencies on EN→UK with 100K UK sentences.

As summarized in Table 5.18, our method fails to perform well on these sets of languages. Although Thai and Lao are very closely related languages, we attribute this result to little subword overlap in their respective vocabularies which degrade the quality of the embeddings. This is because Lao's writing system is developed phonetically whereas Thai writing contains many silent characters. Considering shared phonetic information while learning the embeddings can alleviate this issue and is an avenue for future work. On the other hand, Amharic and Tigrinya, while sharing a decent amount of vocabulary, use different constructs and function words (Kidane et al., 2021) leading to a very noisy psuedo-parallel corpus.

### 5.2.5 Related Work

Early work addressing translation involving language varieties includes rule-based transformations (Altintas and Cicekli, 2002; Marujo et al., 2011; Tan et al., 2012) which rely on language specific information and expert knowledge which can be expensive and difficult to scale. Recent work to address this issue only focuses on cases where parallel data do exist. They include a combination of word-level and character-level MT (Vilar et al., 2007; Tiedemann, 2009; Nakov and Tiedemann, 2012) between related languages or training multilingual models to translate to/from English to different varieties of a language (e.g., Lakew et al. (2018) work on Brazilian–European Portuguese and European–Canadian French). Such parallel data, however, are typically unavailable for most language varieties.

Unsupervised translation models, which require only monolingual data, can address this limitation (Artetxe et al., 2018; Lample et al., 2018a; Garcia et al., 2020, 2021). However, when even *mono*lingual corpora are limited, unsupervised models are challenging to train and are quite ineffective for translating between unrelated languages (Marchisio et al., 2020). Considering varieties of a language as writing styles, unsupervised style transfer (Yang et al., 2018; He et al., 2020) or deciphering methods (Pourdamghani and Knight, 2017) to translate between different varieties have also been been explored but have not been shown to perform well, often only reporting BLEU-1 scores since they obtain BLEU-4 scores which are closer to 0. Additionally, all of these approaches require simultaneous access to data in all varieties during training and must be trained from scratch when a new variety is added. In contrast, our presented method allows for easy adaptation of SRC→STD models to any new variety as it arrives.

Considering a new target variety as a new domain of STD, unsupervised domain adaptation methods can be employed, such as finetuning SRC→STD models using pseudo-parallel corpora generated from monolingual corpora in target varieties (Hu et al., 2019; Currey et al., 2017). Our proposed method is most related to this approach; but while these methods have the potential to adapt the decoder language model, for effective transfer, STD and TGT must have a shared vocabulary which is not true for most language varieties due to lexical, morphological, and at times orthographic differences. In contrast, our proposed method makes use

| | |
|---|---|
| Source | And we never think about the hidden connection |
| Reference | Та ми ніколи не думаємо про приховані зв'язки |
| PIVOT | І ми ніколи не дуємо про приховану зв'язку. (And we never think about a hidden connection.) |
| SOFTMAX | Я ніколи не думав про прихований зв'язок. (I never thought of a hidden connection.) |
| LANGVARMT | І ми ніколи не думаємо про прихований зв'язок. (And we never think about a hidden connection.) |
| Source | And yet, looking at them, you would see a machine and a molecule. |
| Reference | Дивлячись на них, ви побачите машину і молекулу. |
| PIVOT | І бачити, дивлячись на них, ви бачите машину і молекулу молекули. (And to see, looking at them, you see a machine and a molecule of a molecule.) |
| SOFTMAX | І так, дивлячись на них, ви бачите машину і молекулу. (And so, looking at them, you see a machine and a molecule.) |
| LANGVARMT | І дивляючись на них, ви побачите машину і молекулу. (And looking at them, you will see a machine and a molecule) |
| Source | They have exactly the same amount of carbon. |
| Reference | Вони мають однакову кількість вуглецю. |
| PIVOT | Таким чином, їх частка вуглецю. (Thus, their share of carbon.) |
| SOFTMAX | Вони мають однакову кількість вуглецю. (They have the same amount of carbon.) |
| LANGVARMT | Вони мають точно таку ж кількість вуглецю. (they have exactly the same amount of carbon) |

Table 5.16: Examples of EN-UK translations generated by LANGVARMT and the best performing baselines.

of cross-variety word embeddings. While our examples only involve same-script varieties, augmenting our approach to work across scripts through a transliteration component is straightforward.

| | |
|---|---|
| Source | I've never heard of this address near here. |
| Reference | ما قط سمعت بهالعنوان ها<br>لمنطقة من قبل. |
| PIVOT | رح يسلمك.<br>(He will hand over.) |
| SOFTMAX | ولا مرة سمعت عن هالعنوان هني.<br>(Not once did I hear this title here)<br>ما سمعت أبداً من هعنوان قريب من هون.<br>(I've never heard from this address near here.) |
| Source | What's the exchange rate today? |
| Reference | شنو السعر اليوم؟ |
| PIVOT | سعر اليوم؟<br>(What's the rate?) |
| SOFTMAX | شنو سعر الصرااليوم؟<br>(What's the exchange rate today?)<br>شو سعر الصرااليوم؟<br>(What's the exchange rate today?) |
| Source | How do I get to that place? |
| Reference | كيبوصل لهالمطرح؟ |
| PIVOT | كيبتنصح؟<br>(How do you recommend?) |
| SOFTMAX | كيي أوصل عالمحل؟<br>(How can I get to the shop?)<br>كيي وصل؟<br>(How can I get there?) |
| Source | Tell me when we get to the museum. |
| Reference | قلي بس نوصل عالمتح |
| PIVOT | رح نروح عالتاني<br>(we will go to the other.) |
| SOFTMAX | احكي ايمتى نوصل عالمتح<br>(Talk when we get to the museum)<br>قلي ايمتى وصلنا للمتح<br>(Tell me when we got to the museum) |
| Source | Please take me to the morning market. |
| Reference | عمول معروخدني على سوق الصبح. |
| PIVOT | رح نطرني.<br>(We'll wait) |
| SOFTMAX | منتاخدني عالسوق الصبح.<br>(You take us to the market this morning.)<br>منل تاخدني عالسوق الصبح.<br>(We prefer you take us to the market at the morning.) |

Table 5.17: Examples of English to Beirut Arabic translations generated by LANGVARMT and the best performing baselines.

## 5.3 Conclusion and Future Work

This chapter introduces a new framework for training text generation models by predicting token representations, treating it as step-wise regression rather than the commonly adopted step-wise classification. We propose new probabilistic loss functions based on the vMF distribution for learning in this framework and validate its efficacy on text-to-text tasks like machine translation and paraphrasing. Further, we presented

| | EN→LO | EN→TI |
|---|---|---|
| SRC→STD | 0.7 | 1.8 |
| SOFTMAX | 1.4 | 2.9 |
| LANGVARMT | 4.5 | 3.8 |

Table 5.18: BLEU scores for English to Lao and English to Tigrinya translation

a transfer-learning framework for rapid and effective adaptation of such models to different varieties of the language under consideration without access to any gold supervision in the target variety. We demonstrated significant gains in BLEU scores across several language varieties, as well improved fairness of such systems across dialectal speakers especially in highly resource-scarce scenarios.

There remain numerous directions of further exploration in this line of work including improving the foundational training framework of predict word vectors to improve training efficiency and generation quality as well as enable better adaptability to different linguistic variations downstream. Beyond this work, which focuses on machine translation, further work is needed to investigate other pertinent tasks such as general-purpose language models, dialogue generation, summarization, and so on.

In the work presented in this chapter, we use pretrained and fixed embeddings (Bojanowski et al., 2017a) which are then separately used to train the model. While this setup helps in adapting the model to related vocabularies, the pretrained word embeddings are not always well suited for generation and can lead to grammatical and semantic errors as we show in our analysis. Future work may learn both representations simultaneously while maintaining the benefits of this setup by, for example, adopting non-constrastive embedding techniques which have shown great promise visual for representation learning (Ermolov et al., 2020; Zbontar et al., 2021). Several other modeling decisions made in this work can be revisited to improve performance and utility. For example, we showed that Euclidean distance is not an effective training objective and perform regression into a spherical embedding space which is a non-Euclidean space. Other non-Euclidean vector spaces may be explored in the future for both lexical and model representations such as hyperbolic spaces (Nickel and Kiela, 2017; Tifrea et al., 2019) for their low-dimension requirements and inherent hierarchical properties especially useful for language data. Further, the embedding function can be parameterized using a neural network instead of a simple table which can take as input information at the character, morphological or phonological level of the words (Chaudhary et al., 2018) as well as use external resources like dictionaries or WordNet (Pappas et al., 2020) allowing it to explicitly model similarities between words not present in text which can be exploited to adapt the models to dialects which do not share the same orthography as the standard variety (Kumar et al., 2021a).

Furthermore, future work may explore these methods to train multilingual language models by exploiting its potential in data efficiency exploiting similarities across languages, and usefulness in transfer across dialects, code-mixed languages, and text domains with varying lexicons. Finally, in this work, while we train autoregressive models via teacher forcing, predicting continuous word representations as a general framework holds tremendous promise for other kinds of generative language models. For example, Budhkar et al. (2019) found this approach to be promising to train generative adversarial networks (Goodfellow et al., 2020) for text generation although limited in performance by fixed word vectors which are not updated while training the model. More recently, this approach has found success in training diffusion models (Ho et al., 2020) for text generation (Li et al., 2022a; Strudel et al., 2022), which until recently only worked for continuous domains. This class of models is especially useful for adding post hoc controllability to language models which is the focus of the next chapter.

# Chapter 6

# Adapting Pre-Trained Models to Generate Language Varieties

*This chapter discusses work previously published in Kumar et al. (2021b) and Kumar et al. (2022b).*

Recent advances in language models (LMs) (Radford et al., 2019a; Devlin et al., 2019; Raffel et al., 2020) trained on large-scale web text corpora have led to great improvements in state-of-the-art on many natural language processing (NLP) tasks including the ability to generate increasingly coherent text (Brown et al., 2020a). Despite having human-level fluency, they are far from reaching human-level communication abilities and can be hard to control for content, context, and intent in communication including controlling for stylistic variations in the output text. This results in unreliable models that lack basic knowledge, hallucinate facts, and discriminate users (Bender et al., 2021; Gehman et al., 2020; Pagnoni et al., 2021). Controlling the characteristics or attributes of the generated text may require architectural modifications (Keskar et al., 2019a; Krause et al., 2020a; Li and Liang, 2021) and fine-tuning the models on attribute-specific corpora (Krishna et al., 2020; Cheng et al., 2020), as we explored in the previous chapter to generate dialects. However, these methods can be computationally challenging to apply to large language models with billions of parameters and even infeasible if multiple attributes or controls are involved as labeled data for each combination of attributes can be difficult to obtain.

Contrasting from the previous chapter where we finetune language generation systems to generate varieties, in this chapter, we present inference algorithms for text generation models that allow controlling the outputs to contain desired variations, *without modifying the model*. For example, given a dialogue generation model, constraining the generated responses to be polite, even though the model was not optimized for politeness during training. Recent works incorporate control in left-to-right decoding by modifying the vocabulary distribution at every step directly using auxiliary classifiers or language models trained on attribute specific corpora (Yang and Klein, 2021; Krause et al., 2020b; Liu et al., 2021; Lu et al., 2021b; Pascual et al., 2021; Liu et al., 2021) or indirectly via backpropagating gradients through model activations (Dathathri et al., 2020). While effective in certain settings, by generating autoregressively these approaches fail to account for global context and hardly generalize beyond a single control or constraint. More importantly, by modifying output probabilities, they end up altering the underlying model distribution (Kumar et al., 2021b). For example, in the case of dialogue generation, the output should not forgo the task (which is to generate an appropriate response) in lieu of politeness.

Towards addressing these concerns, in this chapter, we propose to generate text non-autoregressively

from pretrained frozen language models trained to perform any generation task—translation, summarization, dialog, prompt completion— while controlling for multiple, potentially competing constraints at the global sequence level, and with a goal to not sacrifice the base model quality. First, we represent each target attribute to control as a differentiable function to minimize. Second, we formulate inference as a multi-objective optimization problem, with maximizing the log-probability of the language model and target attributes functions as objectives. Since language is discrete with a potentially large vocabulary, this combinatorial optimization problem can be prohibitively expensive to solve. To make decoding feasible, we relax it to a continuous optimization problem which allows us to use gradient-based methods considering output tokens as parameters while keeping the language model's parameters fixed—iteratively transforming an output sequence initialized randomly into a desired output.

Based on the types of relaxation and optimization algorithms we propose, this chapter is divided into two parts. First, we explore representing each output token as a simplex on the target vocabulary (Hoang et al., 2017) performing simple gradient decent considering each token distribution as parameters generating one output per input. Second, taking inspiration from the work presented in the previous chapter, we represent each token as a low-dimensional vector using its non-contextual model embedding and generalize optimization to a *sampling* algorithm to generate multiple diverse outputs from the models. We achieve that by interpreting the objective as an energy function and extending gradient descent to a gradient-based Markov Chain (Brooks et al., 2011).

## 6.1 MuCoCO: Constrained Decoding as Multi-Objective Optimization

For a given language generation task, let $\mathcal{G}$ model the conditional probability $P(\mathbf{y}|\mathbf{x};\theta)$ of an output sequence $\mathbf{y} = y_1, \ldots, y_N$, given the input sequence $\mathbf{x} = x_1, \ldots, x_M$. $\mathcal{G}$ can be parameterized using any differentiable architecture (Hochreiter and Schmidhuber, 1997; Vaswani et al., 2017b) and trained with any loss function (Edunov et al., 2018; Kumar and Tsvetkov, 2019). Traditionally, given an input $\mathbf{x}$, decoding from such a model involves finding output(s) $\mathbf{y} \in \mathcal{Y}$ which admit a high probability under $P$. In most cases, it is formulated as finding the highest probability or the lowest negative log-probability sequence, $\mathbf{y}^* = \arg\min_{\mathbf{y}\in\mathcal{Y}} -\log P(\mathbf{y}|\mathbf{x})$. Here $\mathcal{Y}$ is the set of all possible output sequences. In practice, searching $\mathcal{Y}$ to find the highest probability generation is intractable as the space of possible sequences grows exponentially with sequence length and has also been shown to produce undesirable solutions (Stahlberg and Byrne, 2019). Hence, traditionally $P$ is factorized over each token $y_n$, where the output is generated left-to-right one token at a time, where the output token at step $n$ is fed as an input to the model at step $n+1$. It typically also involves different search or sampling strategies such as beam search, top-k sampling (Fan et al., 2018), and nucleus sampling (Holtzman et al., 2020), among others (Meister et al., 2023; Wiher et al., 2022).

In this work, given $\mathcal{G}$ and an input sequence $\mathbf{x}$, we are interested in finding an output sequence $\mathbf{y}$ that not only maximizes the output probability but also optimizes multiple objectives defined over $\mathbf{x}$ and $\mathbf{y}$. More formally, we seek to find a $\mathbf{y}$ that minimizes all of the following objectives

$$\mathbf{y}^* = \arg\min_{\mathbf{y}\in\mathcal{Y}}(-\log p(\mathbf{y}|\mathbf{x}), f_1([\mathbf{x}],\mathbf{y}), \ldots, f_u([\mathbf{x}],\mathbf{y})) \tag{6.1}$$

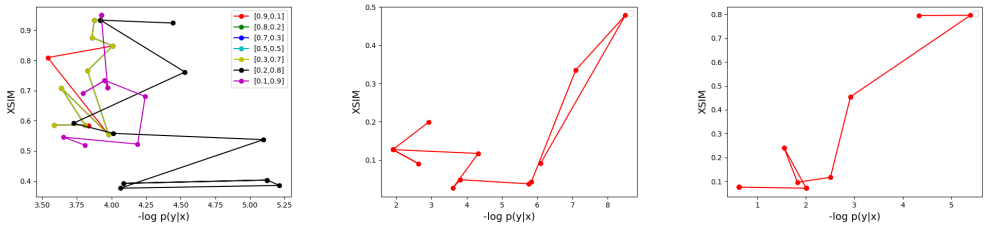
Here each $f_i : ([\mathbf{x}],\mathbf{y}) \to \mathbb{R}$ is a function defined either (a) only over the output sequence $\mathbf{y}$, for example, the negative log-probability of an attribute (e.g., formality or toxicity) classifier we want the output sequence to satisfy or (b) defined over both the input and output sequence, for example, semantic similarity between $\mathbf{x}$ and

$\mathbf{y}$ (Reimers and Gurevych, 2019). We assume all $f_i$ are differentiable and are defined such that a lower value of $f_i$ implies that the output better satisfies the constraint. This is a multi-objective optimization with several possible solutions.

Since there are many objectives to minimize, a left-to-right decoding strategy like beam search or sampling will simply not work due to several reasons. First, the objectives $f_i$ are sentence-level and hard to define accurately only on generated left-context. Even if we are able to define them, as we add more objectives t'his process becomes very computationally expensive. Following prior work (Hoang et al., 2017; Qin et al., 2020), we formulate this as a continuous optimization process instead of a standard discrete one and then use standard algorithms for continuous optimization (like gradient descent) for decoding. We maintain a soft-representation of the sequence $\mathbf{y}$, $\tilde{\mathbf{y}} = (\tilde{y}_1, \ldots, \tilde{y}_n)$, where each $\tilde{y}_k \in \Delta_V$ is a simplex over the target vocabulary of size $V$, representing the probability of the $k$-th token. To decode a sentence, we initialize each $\tilde{y}_i$ uniformly over $V$, and treat the entire output sentence as the parameters for gradient descent keeping the parameters of $\mathcal{G}, f_i$ fixed. After gradient descent has converged, we generate discrete text by selecting the token with the highest probability in $\tilde{y}_k$. We provide more details on the optimization procedure in §6.1. To make optimization feasible, a multi-objective problem generally yields itself to the following formulation:

$$\arg\min_{y} -\alpha \log p(\mathbf{y}|\mathbf{x}) + \sum_{i=1}^{u} \lambda_i f_i([\mathbf{x}], \mathbf{y}) \tag{6.2}$$

for some statically or dynamically computed weights $\lambda_i$'s, where $0 < \alpha, \lambda_i < 1$, and $\alpha + \sum_i \lambda_i = 1$. Although this weighted summation formulation is intuitively appealing, it typically requires an expensive grid-search over the various scalings or use of a heuristic (Kendall et al., 2018; Chen et al., 2018; Guo et al., 2018). Furthermore, this formulation by definition assumes a trade-off between the different objectives by essentially assigning an importance weight to each of them. This problem is further exacerbated when different objectives have widely varying scales[1] with smaller scale objectives just getting ignored. More concretely, a multi-objective formulation as we define in (6.1) admits several possible "optimal" solutions also known as the Pareto set (Debreu, 1954). The image of the Pareto set is called the Pareto front. Since we define all objectives using neural networks, the Pareto front in our case is non-convex, where linear combinations of objectives are shown to be unsuccessful in finding good solutions (Lin et al., 2019, 2021; Degrave and Korshunova, 2020) (see figure 6.1 for an example).




(a) Linear combinations. (b) MUCOCO without damp. (c) MUCOCO

Figure 6.1: Loss curves for gradient descent for different configurations for an example of machine translation with a cross-lingual semantic similarity constraint (XSIM $< 0.15$). For each experiment, we do 100 steps of gradient descent (for clarity, we plot the loss values for every 10 steps). See §6.1.3 for detailed results. Left: In all cases one of the objectives is favored while the other fails to decrease. Middle: We observe fluctuations in the two losses. Right: The losses decrease much more smoothly leading to a better minimum.

[1]For example, classifier log-probabilities are in $(0, \inf)$ while sentence similarities usually lie in [0,1].

Ideally, our goal is a tunable optimization algorithm that finds solutions on the Pareto front, i.e., every solution on the Pareto front should have a hyperparameter value for which the optimization algorithm finds that solution. In order to achieve this, we reframe our optimization problem as a Lagrangian optimization problem instead. We choose one of the losses as the primary objective and consider other losses as constraints. The goal is to minimize the primary loss subject to the secondary losses, each below a threshold value. More formally,

$$\arg\min_{\mathbf{y}} -\log P(\mathbf{y}|\mathbf{x}) \text{ subject to}$$
$$f_i([\mathbf{x}], \mathbf{y}) \leq \epsilon_i, i \in \{1, \cdots, u\}.$$

Here $\epsilon_i$ are tunable hyperparameters whose values' change can result in different solutions on the Pareto front. This formulation leads to an intuitive interpretation of the decoding process that the generated text from the model $\mathcal{G}$ should satisfy the constraints while being as faithful to the primary objective as much as possible. For example, defining $f_i(\mathbf{y}) = p(a|\mathbf{y})$ as the probability of a desired attribute $a$ in $\mathbf{y}$ leads to a natural threshold of $f_i(\mathbf{y}) > 0.5$.[2] Consequently, the Lagrangian we end up with looks similar to our original total loss linearly combined as in (6.2) given by

$$\mathcal{E}(y, \lambda_1, \ldots, \lambda_u) = -\log p(\mathbf{y}|\mathbf{x}) - \sum_{i=1}^{u} \lambda_i(\epsilon_i - f_i([\mathbf{x}], \mathbf{y})) \tag{6.3}$$

where $\lambda_i \geq 0$ now denote Lagrange multipliers (which are not predefined), and an optimal output $\mathbf{y}^*$ can be obtained as $\mathbf{y}^* = \arg\min_{\mathbf{y}} \max_{\lambda_i \geq 0} \mathcal{E}(\mathbf{y}, \lambda_i)$. $\mathcal{E}$ is also referred to as "energy" throughout this chapter. However, the traditional method of solving this dual function to find $\lambda_i$'s again can lead to a linear trade-off between the various objectives. When the Pareto front is non-convex as it is in our case, with gradient-descent, the constraints can be ignored and we still cannot always find optimal solutions by tuning $\epsilon_i$ (Platt and Barr, 1988).

**Modified Differential Method of Multipliers** The fundamental issue in both linear combination of objectives and solving the dual is that $\lambda_i$'s are fixed and do not change during optimization. Following prior work on differential method of multipliers (Platt and Barr, 1988), we propose to use a single gradient descent to optimize for both Lagrangian multipliers and $\mathbf{y}$ simultaneously as follows:

$$\mathbf{y}^{(t)} = \mathbf{y}^{(t-1)} - \eta_1 \nabla_{\mathbf{y}} \mathcal{E} \tag{6.4}$$
$$\lambda_i^{(t)} = \lambda_i^{(t-1)} + \eta_2 \nabla_{\lambda_i} \mathcal{E} \tag{6.5}$$

We follow the gradient of $\mathcal{E}$ downwards for $\mathbf{y}$ (descent) and upwards for the multipliers (ascent) while making sure that the multipliers remain positive (by setting the multipliers to 0 whenever they become negative). Intuitively, this algorithm works by increasing the value of the multiplier with each gradient step as long as the constraint is violated. But when the constraint is suddenly satisfied and the multiplier is still large, it might take a number of gradient steps before the gradient descent pushes it to 0, thus causing the solution to be pushed further away from the constraint. As soon as the multipliers become 0 (or negative), the constraint is ignored and the process continues. However, when the optimization hits the constraint again, this whole cycle repeats, resulting in "oscillations". We introduce a dampening parameter to each of the multipliers to reduce

[2] For a well-calibrated $f_i$, an even higher threshold could be used for inducing highly indicative features of $a$ in $\mathbf{y}$.

these oscillations (again following Platt and Barr (1988)) and update the Lagrangian as follows:

$$\mathcal{E}(\mathbf{y}, \lambda_i) = -\log p(\mathbf{y}|\mathbf{x}) - \sum_{i=1}^{u} (\lambda_i - \zeta_i)(\epsilon_i - f_i([\mathbf{x}], \mathbf{y})), \tag{6.6}$$

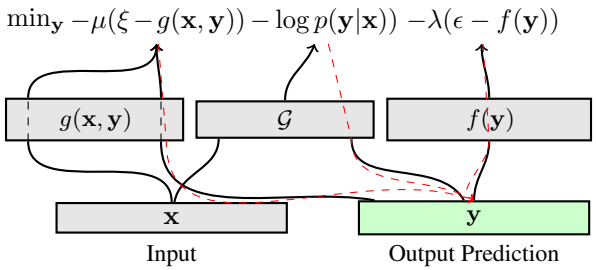



Figure 6.2: MUCOCO architecture. At each step, only the output sequence $\mathbf{y}$ is updated by receiving gradients from the primary objective of the base text generation model $\mathcal{G}$ as well as the constraints $f$ and $g$, corresponding to arbitrary text attributes to control for at decoding time. Any number of differentiable constraints can be incorporated. Black arrows indicate forward pass while the red dashed arrows indicate the backward pass. The parameters of all the objectives remain frozen (shown in gray).

where $\zeta_i = d * \texttt{stop-gradient}(\epsilon_i - f_i([\mathbf{x}], \mathbf{y}))$ and $d$ is a hyperparameter. $d$ does not affect the final $\mathbf{y}$, just how quickly the algorithm converges to it (We use $d = 1$ in all experiments). $\texttt{stop-gradient}(\cdot)$ indicates that the argument is detached from the computational graph and does not contribute to the gradient computation. When a constraint is not satisfied ($\epsilon_i - f_i([\mathbf{x}], \mathbf{y}) < 0$, hence $\zeta_i < 0$), the dampening parameter $\zeta_i$ being negative incurs higher penalty on the violation than when not using any dampening, without actually increasing the value of $\lambda_i$ too much. But when the constraint is satisfied, it helps quickly reduce the value of penalty being incurred on the constraint while the multiplier converges to 0.

**Optimization: Exponentiated Gradient Descent** Our goal is to generate a sequence of discrete symbols $\mathbf{y} = y_1, \ldots, y_T$, where $y_k$ is from the target vocabulary. To make continuous optimization like gradient descent feasible, we adopt a soft-relaxation (Hoang et al., 2017) to represent each $y_k$ as a probability simplex, $\tilde{y}_k \in \Delta_V$ (i.e. $0 \leq \tilde{y}_{kl} \leq 1$ and $\sum_{l=1}^{|V|} \tilde{y}_{kl} = 1$). Intuitively, it gives the probability of each token in the vocabulary. To compute the loss $\mathcal{E}$ during forward pass, we first convert $\tilde{y}_k$ to a one-hot vector $\hat{y}_k$ via a straight through estimator (Bengio et al., 2013). This allows gradients to be applied to $\tilde{y}_k$ during the backward pass. More formally, $\hat{y}_k = \texttt{one-hot}(\arg\max \tilde{y}_k) - \texttt{stop-gradient}(\tilde{y}_k) + \tilde{y}_k$. During the forward pass, the input embedding tables corresponding to $\mathcal{G}$ and each of the constraints' models receive a one-hot vector $\hat{y}_k$ at each step $k$, and the input embedding is computed as a weighted-sum of the embedding weights. But in the backward pass, the gradients are applied to $\tilde{y}_k$.[3]

This relaxation, however, adds another constraint to the objective $\mathcal{L}$ that each parameter $\tilde{y}_k$ should be a simplex. We use exponentiated gradient descent (Kivinen and Warmuth, 1997; Hoang et al., 2017) to solve this problem which modifies the gradient-descent update shown in (6.4) as: $\tilde{y}_k^{(t)} \propto \tilde{y}_k^{(t-1)} \exp(-\eta_1 \nabla_{\tilde{y}_k} \mathcal{L})$. After every descent step, $\tilde{y}_k^{(t)}$ is normalized to make it a simplex.

[3]Unlike prior work (Hoang et al., 2017; Qin et al., 2020; Song et al., 2020), we do not feed $\tilde{y}_i$ directly to the model as in our early experiments we found that it leads to slow convergence.

**Preventing adversarial solutions: Annealing the thresholds** Finally, it is well known that most neural network based models are not robust to noise and in fact gradient-based methods have been used to generate adversarial examples for text classifiers (Song et al., 2020). We find in our early experiments that using these models to define constraints can also lead to such cases where the constraints are rapidly satisfied but the generated sentences are disfluent. To prevent this issue, we introduce an annealing schedule (Paria et al., 2020) during the gradient descent where we start with relaxed thresholds $\epsilon_i, \xi_j$ such that they are all satisfied and only the primary loss $-\log p(\mathbf{y}|\mathbf{x})$ is active. As the optimization progresses, we gradually decrease the value of the thresholds causing the constraints to get violated resulting in the optimization gradually shifting to updating $\mathbf{y}$ to satisfy them. The exact schedule we use is described in the next section.

The final decoding algorithm we use in all our experiments is described in the Appendix algorithm 2. We call the algorithm MUCOCO for text generation with **mu**ltiple **co**nstraints via **c**ontinuous **o**ptimization (see figure 6.2).

**Algorithm 2:** MUCOCO: detailed decoding algorithm

**Input:** input sequence $\mathbf{x}$, output length $L$, base model $\mathcal{G}$, attribute functions $f_i$ and $g_j$ and their respective initial and final thresholds, threshold update schedule, step sizes $\eta_1$, $\eta_2$;

**Result:** output sequence $\mathbf{y}$

For all $k \in \{1, \ldots, L\}$, initialize $\tilde{\mathbf{y}}_k^0$ uniformly over $\Delta_V$;

For all $i \in \{1, \ldots u\}$ and $j \in \{1 \ldots v\}$, initialize $\lambda_i^0, \mu_i^0$ as 0 and the thresholds $\epsilon_i^0, \xi_j^0$ with the given values ;

**for** $t = 1, \ldots,$ MAXSTEPS **do**

`// forward pass`

for all $k$, compute $\hat{y}_k = \texttt{one-hot}(\arg\max \tilde{y}_k)$ and compute the loss $\mathcal{L}$ (using (6.6));

`// backward pass`

for all $k, i$ and $j$, compute $\nabla_{\tilde{y}_k}^{t-1} = \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{y}}_k}$, $\nabla_{\lambda_i}^{t-1} = \frac{\partial \mathcal{L}}{\partial \lambda_i}$, $\nabla_{\mu_j}^{t-1} = \frac{\partial \mathcal{L}}{\partial \mu_j}$;

`// Update the parameters`

update $\tilde{y}_k^{(t+1)} \propto \tilde{y}_k^{(t)} \exp(1 - \eta_1 \nabla_{\tilde{y}_k} \mathcal{L})$;

update $\lambda_i^t = \max(0, \lambda_i^{t-1} + \eta_2 \nabla_{\lambda_i} \mathcal{L})$, and $\mu_i^t = \max(0, \mu_i^{t-1} + \eta_2 \nabla_{\mu_i} \mathcal{L})$;

update $\epsilon_i^t, \xi_j^t$ following the threshold update schedule

**end**

**return** $\arg\min_t \{-\log p(\tilde{\mathbf{y}}^{(t)}|\mathbf{x}) : \forall i, f_i(\tilde{\mathbf{y}}^{(t)}) \leq \epsilon_i, \forall j, g_j(\mathbf{x}, \tilde{\mathbf{y}}^{(t)}) \leq \xi_j\}$;

### 6.1.1 Experimental Setup

We evaluate MUCOCO on the following controlled generation tasks[4]: reinforcing target style in text generated by a style transfer model §6.1.2 and adding formality to a machine translation model (§6.1.3). Additionally, we conduct a qualitative analysis of rewriting a product review to adhere to multiple expected attributes like formality, sentiment magnitude, and age group of the author (§6.1.4). These tasks include constraints corresponding to both expected attributes in the target sentence (like formality) as well as both source and target sentences (like semantic similarity) with up to 6 constraints per task.

**Implementation Details** For a given sentence length $T$, we initialize each simplex $\tilde{y}_1, \ldots, \tilde{y}_T$ uniformly over the vocabulary. We use exponentiated descent learning rate of $\eta_1 = 50$ for $\mathbf{y}$ and ascent learning rate

[4]This algorithm is also interesting to evaluate in a unconstrained setting by simply maximizing the target probability. This setting has been extensive explored in Hoang et al. (2017). Confirming their findings, in our initial exploration with machine translation datasets, we found this setup to perform similarly to beam search.

of $\eta_2 = 2.0$ for the multipliers, and run the optimization for $100$ steps. Given all intermediate solutions $\mathbf{y}^{(t)}$, we choose the one which satisfies the constraints and has the minimum value of the primary objective. For each constraint, we use the following annealing schedule: we start with an initial value and linearly decrease it at step $40$ until it reaches the desired value at step $80$, after which we keep it constant. Additionally, since the length of the target sequence is not known in advance, we first greedily decode from $\mathcal{G}$ till the end-of-sentence token is generated resulting in a sequence of length $L$. We then use our approach for each $T \in \{L-5, \ldots, L+5\}$ and choose the one which (a) satisfies all the constraints and (b) has the minimum value of the primary objective. However, this optimization objective is highly non-convex and may get stuck in a local minimum where constraints are not satisfied. If none or partial constraints are satisfied, we choose the output based on (b).

### 6.1.2 Style Transfer

We begin with a style-transfer task, a task aiming to faithfully and fluently rewrite a given sentence such that a desired writing style is reflected in the generation. This task has been widely studied (Hu et al., 2017; Shen et al., 2017; Krishna et al., 2020, among others) and differs from related tasks like sentiment transfer (Sudhakar et al., 2019; Lample et al., 2019b; Li et al., 2018) where flipping the sentiment usually comes at the cost of changing meaning.

Style transfer is usually evaluated across three dimensions: (1) does the output sentence conform to the expected style; (2) does the output sentence preserve the input's meaning; and (3) is the generated sentence fluent. Most prior work in style transfer focused on devising training objectives serving as proxy for the desired outcomes, for example, back-translation (Prabhumoye et al., 2018; Lample et al., 2019b) or paraphrasing (Krishna et al., 2020) for content preservation and language modeling for style and fluency. But depending on training algorithm and available data, there is often an observed trade-off between transfer and content-preservation (Prabhumoye et al., 2018; Lample et al., 2019b). To that end, we add the desired attributes via explicit constraints when decoding from an existing style transfer model.

More specifically, we consider the task of informal to formal transfer (Rao and Tetreault, 2018) with the state-of-the-art unsupervised model STRAP from Krishna et al. (2020). This model is trained in an unsupervised fashion by (1) generating a pseudo-parallel corpus by paraphrasing each formal sentence in the training set (which results in a demotion of stylistic attributes), and (2) training an inverse-paraphrase model to translate paraphrases back to the original formal style. At test time, given an informal input sentence $\mathbf{x}$, the model first generates its paraphrase $\mathbf{z}$, then using an inverse-paraphrase model to generate the output $\hat{\mathbf{y}}$. We train this model by fine-tuning GPT2 (345M) (Radford et al., 2019a) with the GYAFC Corpus (Entertainment/Music domain; around 50K formal sentences) (Rao and Tetreault, 2018) and evaluate it on the provided test set containing 1312 informal sentences. Krishna et al. (2020) report best results with greedy decoding. In MUCOCO we modify the decoding algorithm by considering the negative log-probability of $\mathbf{y}$ given $\mathbf{z}$ according to the model as the primary objective, and incorporate the following constraints:

**Formality**: We train a binary classifier $p_{\text{FORMAL}}(\mathbf{y})$ by fine-tuning GPT2 on the same GYAFC training corpus, following default hyperparameter choices provided in HuggingFace (Wolf et al., 2020). This classifier outputs the formality probability of a sentence $\mathbf{y}$. We add this output as a constraint to the decoder as $-\log(p_{\text{FORMAL}}(\mathbf{y})) < -\log(0.5)$. In other words, the constraint is satisfied if the classifier assigns at least 0.5 probability of the output $\mathbf{y}$ being formal. We initialize the threshold to $10.0$ which is later annealed to $-\log(0.5)$.

**Semantic Similarity**: Since the baseline style-transfer model takes as input the paraphrase $\mathbf{z}$ and not the

original text $\mathbf{x}$, it is susceptible to losing some of the original content in $\mathbf{x}$ while generating $\mathbf{y}$. To ensure content preservation we incorporate two kinds of objectives:

(1) $\text{USIM}(\mathbf{x}, \mathbf{y}) = \text{cosine}(M(x), M(y))$ (Reimers and Gurevych, 2019) where $M$ outputs a continuous vector representation of a given sentence. Similarity between $\mathbf{x}$ and $\mathbf{y}$ is measured by cosine similarity of their respective representations. This model is parameterized by GPT2(345M) (Radford et al., 2019a). $M(x)$ is obtained by first feeding $\mathbf{x}$ to the model and then mean pooling all the output representations. This model originally presented in Reimers and Gurevych (2019) is trained in a Siamese fashion on BERT Liu et al. (2019) but is easily extensible to any LM architecture. We adapt it to GPT2 as follows:

- First, we fine-tune $M$ = GPT2 on the combination of SNLI and MNLI (Williams et al., 2018) corpora which are both designed for training natural language inference model and intended to capture semantics. Each corpus contains pairs of sentencse with one of the three annotations: inference, contradiction or neutral. For each input sentence $(\mathbf{s}_1, \mathbf{s}_2)$, the model is trained as with classification objective with the final logits computed as $W[M(\mathbf{s}_1), M(\mathbf{s}_2), |M(\mathbf{s}_1) - M(\mathbf{s}_2)|]$, where $W$ is a trainable parameter. In other words the three vectors as shown are concatenated and multiplied with a weight matrix. We train this for 1 epoch on the combined corpora.

- Second, we continue fine-tuning the $M$ trained so far on the STS corpus which consists of pairs of sentences annotated with real numbers in $[-1, 1]$ indicating their semantic similarity. We train on this corpus with a mean-square-error loss between $\text{cosine}(M(\mathbf{s}_1), M(\mathbf{s}_2))$ and the given score.

Details of training $M$ can be found in Reimers and Gurevych (2019) where this model is shown to perform competitively on STS benchmarks (Williams et al., 2018).

(2) $\text{WMD}(\mathbf{x}, \mathbf{y})$ takes as input bags of word embeddings of the two sentences and computes the Word Mover's Distance between them (Kusner et al., 2015). This distance is computed by solving a linear program. We adapt the alternating optimization procedure described in (Kumar et al., 2017) to make this loss differentiable through the program. Intuitively, while USIM computes similarity between sentences taking context into account, it can be less robust to certain missing or repeating tokens, whereas WMD measures lexical overlap between input sentences acting as a proxy for coverage. Given two bags of words, $x = \{x_1, \ldots, x_n\}$ and $y = \{y_1, \ldots, y_m\}$, and an embedding table $\mathbf{e}$, we define word mover's distance between $\mathbf{x}$ and $\mathbf{y}$ as

$$\text{WMD}(\mathbf{x}, \mathbf{y}) = \min \sum_{i=1,j=1}^{m,n} T_{ij} d_{ij} \text{subject to}$$
$$\sum_{i}^{n} T_{ij} = \frac{1}{m}$$
$$\sum_{j}^{m} T_{ij} = \frac{1}{n}$$

where we define $d_{ij} = 1 - \cos(\mathbf{e}(x_i), \mathbf{e}(y_j))$. Given fixed inputs $\mathbf{e}(x_i)$ and $\mathbf{e}(y_j)$, WMD can easily be computed using linear program solver [5]. To backpropagate through this objective. We use the following steps following Kumar et al. (2017):

1. During the forward pass, we obtain $\hat{\mathbf{y}}$ as indicated in algorithm 2 and compute word embeddings for both the input $\mathbf{x}$ and the prediction $\hat{\mathbf{y}}$. Using the linear program solver, we compute $\text{WMD}(\mathbf{x}, \hat{\mathbf{y}})$ as well the proportions $T_{ij}$

[5] We solve it using the python library POT: https://pythonot.github.io/

2. During the backward pass, we keep the $T_{ij}$ fixed which removes the constraints from the WMD computation as described making it differentiable allowing gradients to flow to update the optimization parameters $\tilde{y}$.

We use the embedding table from USIM model as $\mathbf{e}$ for this constraint.

To compute the thresholds for constrained optimization, we compute the average value of the two functions on the development set in the same corpus. We use USIM $\leq 0.15$ and WMD $\leq 0.4$ as the final constraints (with initial threshold values of 2.0 for each).

**Baselines and Evaluation Metrics** We compare MUCOCO with the following baselines:

NO-CONSTRAINTS: We decode directly from the model greedily without any constraints. This replicates the best result reported by Krishna et al. (2020). We do not use continuous optimization to do unconstrained decoding as it has been shown to perform similarly to left-to-right decoding in prior work (Hoang et al., 2017).

FUDGE: Introduced by Yang and Klein (2021), this method decodes in an autoregressive manner. It modifies the output vocabulary distribution at every step by interpolating the language model probability with that of a formality classifier. This classifier is trained to predict the probability of entire sentence being formal given only a prefix (we train it similarly to $p_{\text{FORMAL}}(\mathbf{y})$ by fine-tuning GPT2). This method only works with categorical features like formality and is not extensible to constraints like semantic similarity. We decode using the hyperparameters recommended in Yang and Klein (2021).

To show the benefits of the constrained optimization setup, we show additional comparisons with a linear combination of objectives in §6.1.4.

Following the baseline model Krishna et al. (2020), we evaluate the generated sentences with the following metrics: (a) **fluency** or grammatical wellformedness measured by the accuracy of a RoBERTa-based classifier model (Liu et al., 2019) trained on CoLA (Warstadt et al., 2018), averaged over all outputs, (b) **transfer**: measured by a RoBERTa-based classifier model (Liu et al., 2019) trained on the GYAFC training corpus, and finally (c) **WSIM** (Wieting et al., 2019), a subword embedding based similarity model trained on a large-scale paraphrase corpus which performs well on STS benchmarks (Cer et al., 2017) as well. We measure this metric both with respect to the input and the provided references.[6] In addition, we also report USIM.

**Results** The style transfer results are summarized in table 6.1. If we only incorporate a formality constraint, we observe that compared to FUDGE our method significantly improves transfer accuracy at the expense of content preservation. Adding semantic similarity constraints on the other hand improves both transfer as well as content preservation with the largest gains achieved when all the constraints are considered together. Qualitative analysis shows that MUCOCO's outputs are typically more fluent and have stronger formality signals, but all of the models are prone to propagating errors from the paraphrasing model (see examples in the Appendix table 6.7).

### 6.1.3 Style-controlled Machine Translation

We now evaluate MUCOCO in the task of formality transfer in machine translation. Given a trained MT model, decoding is often done using beam search and the highest probability beam candidate is chosen as the final output. Prior work has explored adding rule-based or heuristic constraints such as length penalty or coverage (Wu et al., 2016) to rerank beam candidates, and adding lexical constraints like penalizing n-gram repetitions (Hokamp and Liu, 2017). In this experiment, we target sentence-level constraints which are otherwise difficult to incorporate in a left-to-right decoding process. Given a trained MT model and the source

[6] Each input sentence has 4 references, we choose the highest WSIM value to compute the average.

| Method | Constraint | Fluency | Transfer | Content Preservation (w.r.t. input) | | Content Preservation (w.r.t. ref) | |
|---|---|---|---|---|---|---|---|
| | | | | WSIM | USIM | WSIM | USIM |
| STRAP | None | 91% | 78% | 0.69 | 0.77 | 0.72 | 0.80 |
| FUDGE | FORMAL(y) | 90% | 85% | 0.71 | 0.77 | 0.73 | 0.81 |
| MUCOCO | FORMAL(y) | 89% | 93% | 0.67 | 0.75 | 0.72 | 0.78 |
| MUCOCO | USIM(x, y) | 92% | 85% | 0.71 | 0.78 | 0.74 | 0.81 |
| MUCOCO | USIM(x, y), WMD(x, y) | 92% | 87% | **0.73** | **0.79** | **0.77** | **0.86** |
| MUCOCO | SIM(x, y), WMD(x, y), FORMAL(y) | **93%** | **92%** | 0.71 | **0.79** | 0.75 | 0.84 |

Table 6.1: Automatic evaluation of fluency, formality transfer, and content preservation for informal-to-formal style transfer models.

text $x$, we use negative log-probability of the translation $y$ under the MT model as our primary objective and incorporate the following constraints for decoding in different combinations:

**Cross-lingual Similarity** Similar to USIM, we define $\text{XSIM}(\mathbf{x}, \mathbf{y}) = \text{cosine}(CM(x), CM(y))$, where $CM$ is a multilingual encoder trained by distilling a monolingual model like $M$ described earlier (Reimers and Gurevych, 2020b). This method was introduced by Reimers and Gurevych (2020b) where they distill a monolingual model such as $M$, to train a cross-lingual model with a small parallel corpus in the languages of interest. Given a parallel sentence pair $(\mathbf{x}, \mathbf{y})$, $CM$ is trained by minimizing the following loss:

$$\mathcal{L}_{\text{XSIM}} = \|M(\mathbf{x}) - CM(\mathbf{x})\|_2^2 + \|CM(\mathbf{x}) - CM(\mathbf{y})\|_2^2$$

That is, representations of the model $M$ and $CM$ for the source sentence are trained to be close together as are the cross-lingual representations of source and target. We parameterize $CM$ also with pretrained GPT2 (345M) (Radford et al., 2019a) model. But GPT2 and the Marian Transformer based MT model Junczys-Dowmunt et al. (2018) we use do not have matching vocabularies. Since the vocabulary of the primary objective and constraints should match for the decoding to work, we replace input word embedding layer of GPT2 with that of the decoder of the translation model before we train the distilled model. We use the TED2020 French-English parallel corpus containing around 400K sentence-pairs to train XSIM and obtain comparable performance as Reimers and Gurevych (2020b) on the cross-lingual STS benchmark. Averaging across the development set, we use $0.2$ as the threshold for the constraint.

**Formality** Unlike style transfer, where the goal is to rewrite text in the desired style, here we seek to generate translations in a desired style directly from an MT model which was not explicitly trained to conform to a specific style. We train a classifier $p_{\text{FORMAL}}(\mathbf{y})$ similarly to one described in previous section by fine-tuning GPT2, but with a different input-embedding table to match the vocabulary of the decoder of the MT model. Again, we use $\log p_{\text{FORMAL}}(\mathbf{y}) > \log(0.5)$ as the constraint.

**Baselines and Evaluation Metrics** We compare MUCOCO with the following two baselines:

BEAMSEARCH: We decode directly from the translation model with a beam search of size $5$.

FUDGE (Yang and Klein, 2021): defined similarly as in the style transfer task but trained to match the decoder vocabulary. As mentioned before, FUDGE only works with categorical attributes like formality and is not easily extensible to constraints like cross-lingual similarity. We use the recommended hyperparameters by Yang and Klein (2021) for decoding.

In Yang and Klein (2021), the authors also compare FUDGE with other baselines such as PPLM (Dathathri et al., 2020) and BEAMSEARCH followed by style transfer. They show that FUDGE vastly outperforms these

| Method | Constraint | BLEU | BertScore | Formality(%) | XSIM |
|---|---|---|---|---|---|
| BEAMSEARCH | None | 42.1 | 0.932 | 0% | 0.85 |
| MUCOCO | XSIM($\mathbf{x}, \mathbf{y}$) | **42.7** | **0.939** | 4% | **0.88** |
| FUDGE | FORMAL($\mathbf{y}$) | 39.2 | 0.922 | 6% | 0.83 |
| MUCOCO | FORMAL($\mathbf{y}$) | 37.5 | 0.913 | **30%** | 0.83 |
| MUCOCO | FORMAL($\mathbf{y}$), XSIM($\mathbf{x}, \mathbf{y}$) | 39.8 | **0.935** | 23% | **0.86** |

Table 6.2: Results of style-controlled machine translation experiments.

baselines. Hence, we only show comparisons with FUDGE in this work. We evaluate along the following metrics: (a) **BLEU** (Papineni et al., 2002): a standard metric for evaluating MT, (b) **BERTScore** (Zhang* et al., 2020): an embedding-based metric which is more robust to changes in surface forms of the words than BLEU. (b) **transfer**: the same RoBERTa-based formality classifier as in our style transfer experiments. We also report XSIM, the constraint we use for decoding.

We experiment with French to English translation with a subset of the OpenSubtitles test set (Lison and Tiedemann, 2016) containing 1360 sentence pairs.[7] This test set contains informal spoken language for both source and target. For the primary objective, we use the Marian Transformer based French (fr) to English (en) model (Junczys-Dowmunt et al., 2018) through Huggingface. We summarize the results of this experiment in table 6.2 with selected examples in the Appendix table 6.8.

**Results** By just using a cross-lingual similarity metric without modifying the model at all, we observe +0.6 improvement in BLEU score as well as BERTScore. Adding a formality constraint leads to considerable gain in formality of the outputs with a drop in BLEU; using both XSIM and FORMAL helps recover some of the drop. The drop in BLEU is unsurprising: since BLEU is a surface-level metric it naturally penalizes the translations that are rephrased to conform to formality constraints. Indeed, as shown in table 6.8, adding a formality constraint leads to changes in sentence structure and vocabulary. On the other hand, we see improvements in BERTScore which is an embedding-based metric, more robust to paraphrasing.

To further validate our results, we conduct a human evaluation of the generated translations. We randomly sample 100 source sentences and their translations generated by beam search and MUCOCO with both FORMAL and XSIM constraints. Two annotators (highly proficient in French and English) to rank the translations on faithfulness (is the source meaning reflected in the translation?) and formality. The options are randomized. We conduct A/B testing to rank translations generated by our method and beam search. We show the annotators the source sentence and two randomized translations (one from beam search and one from our method). We ask them to choose one of the four options: **1**: the first translation is both faithful and formal while the second is not, **2**: the second translation is both faithful and formal while the second is not, **3**: both are faithful and formal, and **4**: both are either unfaithful or informal or both. On the translation pairs where both annotators agree (79 out of 100), the ones generated by our method were favored by annotators 37% percent of the time, while beam search translations were favored only 18% of the time, and 21% translations were equally favored.

### 6.1.4 Discussion and Analysis

**Linear combination of objectives** In figure 6.1b, we gave a motivating example of why linear combination of objectives leads to some of objectives getting ignored. In table 6.3, for one constraint USIM, we vary the

[7] We create this subset by filtering the original test set to contain only sentence pairs for which beam search translations are classified as informal.

| Weights | | Fluency (%) | Transfer (%) | wsim (w.r.t. input) | wsim (w.r.t. ref.) |
|---|---|---|---|---|---|
| $-\log p(\mathbf{y}\|\mathbf{x})$ | USIM | | | | |
| 0.5 | 0.5 | 91% | 77% | 0.70 | 0.68 |
| 0.3 | 0.7 | 90% | 79% | 0.72 | 0.67 |
| 0.1 | 0.9 | 85% | 62% | 0.77 | 0.73 |
| 0.05 | 0.95 | 76% | 60% | 0.81 | 0.76 |
| 0.01 | 0.99 | 30% | 58% | 0.85 | 0.82 |

Table 6.3: Automatic evaluation of fluency, formality transfer, and content preservation for informal-to-formal style transfer models using a linear combination of two objectives ($-\log p(\mathbf{y}|\mathbf{x})$ and $\text{USIM}(\mathbf{x}, \mathbf{y})$ with different weights. Since USIM lies in $[0, 1]$, it gets ignored if its weight is low, however increasing its weight compromises the fluency.

weights of the linear combination and show that to indeed be the case.

**Simultaneously controlling several attributes** One of the main advantages of our proposed approach is its flexibility to introduce any number of constraints (as long as they are differentiable) to the decoding objective. To illustrate this advantage we consider the following problem: given a sentence annotated with following attributes: age group of the author, formality, and sentiment magnitude, rewrite it such that any chosen combination of the attributes are modified while keeping the others fixed and the content preserved (Logeswaran et al., 2018; Lample et al., 2019b). For our primary objective, we use a inverse-paraphrasing model as defined in §6.1.2 which we train on a corpus of Yelp Reviews[8] (Prabhumoye et al., 2018). First, we paraphrase each sentence in the corpus as described in Krishna et al. (2020) creating a pseudo-parallel corpus (of reviews and their paraphrases) and train $\mathcal{G}$ as an inverse-paraphrase model to translate the paraphrases back to the original reviews. We use USIM and WMD for semantic similarity constraints and three classifiers for (a) age group of the author (binary; $< 30$ years or $> 30$ years); (b) formality of the review (binary: informal or formal); (c) sentiment magnitude (five-class classifier ratings of 1 to 5). Here we focus on sentiment amplification rather than transfer. That is, changing the 4-star rating of an input to 5 (or 2 to 1). All the classifiers are trained by finetuning GPT2[9] on the following corpora: **Age**: we use the NUFA corpus (Huang and Paul, 2019) consisting Yelp Restaurant Reviews with 300K sentences per age group (greater than 30 years, and less than 30 years) in the training set. Our classifier achieves an accuracy of $\sim$80% on a balanced test set of 10K sentences. **Formality**: we use GYAFC corpus as described in §6.1.2 for this constraint (with an accuracy of around 92%) on the provided test set. **Sentiment**: we collect Yelp restaurant reviews using scripts provided by Lample et al. (2019b)[10] with a rating from 1 to 5 star. We subsample from this corpus to train our 5-class classifier on 100K reviews per rating obtaining a classification accuracy of around 75% on a held-out test set also sampled from the same corpus.[11] Table 6.4 shows examples of generated sentences with different combinations of attribute values. We do not focus on sentiment transfer in this setting (e.g. changing a 1-star review to 5-star review) because it also changes the meaning of the utterance making semantic similarity and sentiment constraints incompatible with each other where satisfying one violates the other.

[8] This corpus is sentence-tokenized and lowercased with 2.2M sentences not labeled for any attributes.

[9] we use Huggingface (Wolf et al., 2020) with recommended hyperparameters for training all classifiers: https://huggingface.co/transformers/v2.0.0/examples.html

[10] https://github.com/facebookresearch/MultipleAttributeTextRewriting/tree/master/data/Yelp

[11] Due to lack of an established benchmark for this task and due to many possible combinations of attributes, we do not report quantitative results.

| **< 30 years, informal, 4-star** | **one big plus : the coffee is always fantastic .** |
|---|---|
| < 30 years, informal, 5-star | the coffee is always great ! |
| < 30 years, formal, 4-star | this coffee is incredibly good. |
| < 30 years, formal, 5-star | the coffee is consistently outstanding! |
| > 30 years, informal, 4-star | the espresso is usually enjoyed . |
| > 30 years, informal, 5-star | the coffee is usually delicious also! |
| > 30 years, formal, 4-star | the espresso is pleasantly delicious, nonetheless. |
| > 30 years, formal, 5-star | the coffee is brewed to excellence. |
| **< 30 years, informal, 2-star** | **i left our meal feeling a little disappointed .** |
| < 30 years, informal, 1-star | worst feeling with this little meal . |
| < 30 years, formal, 2-star | i felt failed and disappointed by this meal . |
| < 30 years, formal, 1-star | i left our meal feeling anguished, betrayed . |
| > 30 years, informal, 2-star | i was a little disappointed ! |
| > 30 years, informal, 1-star | this meal bummed me out ! |
| > 30 years, formal, 2-star | i felt unsatisfied by this meal. |
| > 30 years, formal, 1-star | i felt complete disappointment after this meal . |

Table 6.4: MUCOCO with multiple constraints and rewriting reviews with different combination of attributes.

| **< 30 years, informal, 2-star** | **i left our meal feeling a little disappointed .** |
|---|---|
| < 30 years, informal, 5-star | i was excited when I left |
| < 30 years, formal, 5-star | i was impeccably good |
| > 30 years, informal, 5star | i was extremely amazing. |
| > 30 years, formal, 5-star | i was exquisite and a bit phenomenal |

Table 6.5: MUCOCO with sentiment transfer instead of amplification. We remove the USIM constraint here as it gets violated. Without that constraint, we observe that while sentiment transfer is achievable, it substantially alters the meaning of the input text.

**Finding other solutions on the Pareto front** As described in §5.1.1, the thresholds $\epsilon, \xi$ are tunable hyperparameters that allow us to find different solutions on the Pareto front. In our experiments so far, based on expected outcomes and how the constraints are defined, we showed results with only one threshold for each constraint. For example, ideally for a well-calibrated text classifier based constraint, this technique should be able to find solutions for any probability as threshold, but most neural-network based classifiers are not well-calibrated and predict the highest probability output as the label, hence a natural threshold for binary-classifiers is a label probability $> 0.5$. In Appendix table 6.6, we show how the outputs change if we modify this threshold to different values. We observe that in most cases the optimization converges to generate words more commonly associated with formality. On the other hand, semantic similarity between two sentences is even harder to define, is less robust to noise, and varies with writing styles of the input sentences. As shown, increasing this threshold for semantic similarity can lead to repetitions and disfluency.

**Speed and memory requirements** The presented decoding algorithm treats each token in the output sequence $\mathbf{y}$ as a parameter for gradient-descent which involves multiple forward and backward passes through the primary generative model $\mathcal{G}$ as well as attribute models. Given an expected sequence length $L$, it optimizes $LV$ parameters which is both memory and time intensive compared to left-to-right decoding. For example, on a single GeForce RTX 2080 Ti (12GB) on which we run all presented experiments, with a batch size of 1, our approach takes approximately 90 minutes on average to decode around 1200 sentences compared to around 20 minutes for FUDGE (Krishna et al., 2020) with a single constraint. For reference, unconstrained

| | |
|---|---|
| Input Sentence | My dad looks like Paul Newman, and my ex looked like king kong |
| Paraphrase | my dad's like Paul Newman, and my ex looks like a king. |
| **Constraints** | **Outputs** |
| FORMAL($\mathbf{y}$) $> 0.5$, USIM($\mathbf{x}, \mathbf{y}$) $< 0.15$ | My dad looks like Paul Newman, and my ex looks similar to King Kong |
| FORMAL($\mathbf{y}$) $> 0.7$, USIM($\mathbf{x}, \mathbf{y}$) $< 0.15$ | My father looks like Paul Newman, and my ex resembles a King Kong |
| FORMAL($\mathbf{y}$) $> 0.9$, USIM($\mathbf{x}, \mathbf{y}$) $< 0.15$ | My father looks like Paul Newman, and my ex possesses the qualities of King Kong approximately |
| FORMAL($\mathbf{y}$) $> 0.7$, USIM($\mathbf{x}, \mathbf{y}$) $< 0.1$ | My dad possesses looks similar to Paul Newman, my ex appears like King King Kong |
| FORMAL($\mathbf{y}$) $> 0.9$, USIM($\mathbf{x}, \mathbf{y}$) $< 0.05$ | My dad possesses the Paul Newman looks similar my ex possesses similar King Kong resemblance |

Table 6.6: Varying thresholds for the constraints to find other solutions on the Pareto front.

beam-search takes 2-5 minutes. Given enough GPU capacity, however, this approach can easily be extended to larger-batches to improve decoding speed. We do not conduct this experiment due to limited available resources. Using 16-bit floating point operations, this can further be improved. Further, given the capability of this approach to incorporate multiple constraints, it can also be used to generate pseudo-parallel data with different attribute combinations which then could be used to train supervised models for attributes for interest resulting in faster models at inference. Finally, memory efficiency can be improved by not optimizing for tokens directly but instead optimize for token embeddings (Kumar and Tsvetkov, 2019). This formulation also removes the requirement for all the models to share a vocabulary. We investigate this formulation in the next section.

### 6.1.5 Examples

#### Style Transfer

We show selected examples from our style-transfer models in Table 6.7. Since the final output $\mathbf{y}$ is generated from the paraphrase $\mathbf{z}$, not the input sentence $\mathbf{x}$, some of the content is at times modified in the final output in decoding without constraints. MUCOCO with content based constraints is able to recover content in some examples and also improve formality of the outputs. But it can still be prone to errors since the content-similarity metrics are not perfect. See §6.1.2 for more details.

#### Style-controlled Machine Translation

Table 6.8 lists few selected examples for inducing cross-lingual similarity and formality constraints in a French to English MT model. We find that inducing formality modifies some of the constructs (like removing contractions: "gonna" to "going to") in the output sentences which are not measured accurately by a surface-level metric like BLEU. See §6.1.3 for more details.

| | |
|---|---|
| Input | all are good movies u cant rate them |
| Paraphrase | all good films you can't rate |
| NO-CONSTRAINTS | All good movies that you can't rate |
| FUDGE(FORMAL) | All good movies you can not rate |
| MUCOCO (FORMAL,USIM,WMD) | All are good movies you can not rate them adequately |
| Input | One guy says to me 'So, where's the fox from Mystic Pizza? |
| Paraphrase | "so, where's the Mystic Pizza guy?" Said one of the men. |
| NO-CONSTRAINTS | One of the men replied , "So , where is the Mystic Pizza guy? |
| FUDGE(FORMAL) | One of the men replied, "Where is the Mystic burger?" |
| MUCOCO (FORMAL,USIM,WMD) | One of the men enquired, "Where is the Mystic Pizza man?" |
| Input | chill out sweetie everything will be fine eventually. |
| Paraphrase | calm down, sweetheart, everything will be fine. |
| NO-CONSTRAINTS | Calm down, sweetheart, everything is going to be alright. |
| FUDGE(FORMAL) | Calm down sweetheart, everything will be alright. |
| MUCOCO (FORMAL,USIM,WMD) | Calm down sweetheart, everything will eventually proceed to be good. |
| Input | what kinda person are you to even ask that ridiculous (er...spell check that...) question? |
| Paraphrase | what kind of fool would ask such a silly question? |
| NO-CONSTRAINTS | What kind of idiot would ask such a dumb question? |
| FUDGE(FORMAL) | What kind of fool would ask such a dull question? |
| MUCOCO (FORMAL,USIM,WMD) | What kind of foolish person would ask such a dull question? |
| Input | I spent my money, enjoyed myself and didn't loose more than I could afford - I won.... |
| Paraphrase | I've spent my money, I've enjoyed it, and I didn't lose anything. |
| NO-CONSTRAINTS | I spent my money, I enjoyed it, and I did not lose anything. |
| FUDGE(FORMAL) | I have spent my money, I have enjoyed it, and I did not lose anything. |
| MUCOCO (FORMAL,USIM,WMD) | I spent my money, did not lose anything more, and it was simply enjoyable. |

Table 6.7: Style transfer examples with different decoding methods and constraints.

## 6.2 MUCOLA: Gradient-Based Constrained Sampling from Language Models

The setup in §6.1 describes a deterministic algorithm. While we show it is useful for text-to-text tasks where usually one output is desired, it has limited use in open-ended tasks like prompt-based generation or dialogue generation where several diverse outputs are possible all with a high likelihood.[12]. In addition, representing each token with a simplex vector of size $|\mathcal{V}|$ can be computationally very expensive and difficult to fit into commonly used GPUs for long sequences (with more than ∼20-30 tokens; see §6.2.4). In this work, we focus on constrained *sampling*—finding output sequences $\mathbf{y}$ that have a high probability under $P$ while minimizing a given set of constraint functions: $\{f_1, \ldots, f_C\}$. That is,

$$\mathbf{y} \sim P(\mathbf{y}|\mathbf{x}; \theta), \text{ subject to } f_i([\mathbf{x}], \mathbf{y}) \leq \epsilon_i \forall i$$

To enable efficient gradient-based sampling from language models, we generalize the non-autoregressive framework in §6.1 to (a) generate multiple samples instead of optimizing for only one deterministic output, (b)

[12]While initialization can be used to add randomness to MUCOCO, we find that it has little to no effect on diversity.

| | |
|---|---|
| Source | Mais il s'agit... il s'agit d'une femme que vous ne connaissez pas. |
| Reference | But this is– This is a woman you don't know. |
| BEAMSEARCH | But this is... this is a woman you don't know. |
| MUCOCO (XSIM) | But this is... this is a woman you don't know. |
| FUDGE(FORMAL) | But this is... this is a woman you do not know. |
| MUCOCO (FORMAL) | But this is... is a woman you do not know. |
| MUCOCO (FORMAL,XSIM) | But this is a woman you do not know. |
| Source | Toi ? Le mec à bananes, exact. |
| Reference | - Who's the banana man, alright. |
| BEAMSEARCH | You, the banana guy, right. |
| MUCOCO (XSIM) | You? the banana guy, right. |
| FUDGE(FORMAL) | You, the banana guy, right? |
| MUCOCO (FORMAL) | Are you the banana guy? |
| MUCOCO (FORMAL,XSIM) | Are you the banana guy? |
| Source | Nous allons les sortir de la d'ici quelques minutes. |
| Reference | We'll have them out in a couple minutes. |
| BEAMSEARCH | We're gonna get them out of here in a few minutes. |
| MUCOCO (XSIM) | We're gonna get them out of here in a few minutes. |
| FUDGE(FORMAL) | We'll get them out of here in a few minutes. |
| MUCOCO (FORMAL) | We will get them out of here. |
| MUCOCO (FORMAL,XSIM) | We will get them out of here in a few minutes. |
| Source | On va prendre la voie aérienne. |
| Reference | We'll take the aerial up. |
| BEAMSEARCH | We're gonna take the airway. |
| MUCOCO (XSIM) | We're gonna take the air route. |
| FUDGE(FORMAL) | We are gonna take the airway. |
| MUCOCO (FORMAL) | We are going to take the air. |
| MUCOCO (FORMAL,XSIM) | We are going take the air route. |
| Source | Mais mon sang ne correspondait pas. |
| Reference | But my blood didn't match. |
| BEAMSEARCH | But my blood wasn't matching. |
| MUCOCO (XSIM) | But my blood didn't match. |
| FUDGE(FORMAL) | But my blood wasn't matched. |
| MUCOCO (FORMAL) | But my blood was not correct. |
| MUCOCO (FORMAL,XSIM) | But my blood did not match. |

Table 6.8: Translation examples with different decoding methods and constraints.

optimize for much smaller intermediate token representations as opposed to their distribution on the entire vocabulary. First, we describe our proposed way to represent tokens followed by how they can facilitate sampling.

**Exploring the token representation space** Instead of relaxing each target token $y_n$ as a soft representation over the vocabulary $\tilde{y}_n \in \mathbb{R}^{|\mathcal{V}|}$, we represent it as $\tilde{e}_n \in \mathbf{E}$. Here $\mathbf{E}$ denotes the embedding table of the underlying language model containing $|\mathcal{V}|$ vectors of size $d \ll |\mathcal{V}|$. We denote this sequence of embeddings as $\tilde{\mathbf{e}} = \{\tilde{e}_1, \ldots, \tilde{e}_N\}$. At an update step $t$, instead of feeding each $\tilde{\mathbf{y}}$ to the model(s) (which are then transformed to an embedding to be fed to the first layer), we directly feed $\tilde{\mathbf{e}}$ to the first layer to compute the energy function,

now defined as a function of embeddings instead of tokens. In the case of deterministic minimization (similar to §6.1), these vectors are updated as

$$\tilde{\mathbf{e}}^t = \text{Proj}_{\mathbf{E}}(\tilde{\mathbf{e}}^{t-1} - \eta \nabla_{\tilde{\mathbf{e}}} \mathcal{E}(\tilde{\mathbf{e}}^{t-1})), \tag{6.7}$$

where $\text{Proj}_{\mathbf{E}}(\hat{e}) = \arg\min_{e \in \mathbf{E}} \|e - \hat{e}\|_2$ denotes a projection operation on the embedding table $\mathbf{E}$. In other words, after every gradient step, we project each updated vector back to a quantized space, that is the embedding table using Euclidean distance as the metric. This projection is done to prevent adversarial solutions.[13] After the optimization is complete, discrete text can be easily obtained by projection, that is the token indices corresponding to each $\tilde{e}_n$ in the embedding table $\mathbf{E}$. This formulation yields the following benefits: (a) For a sequence of length $L$, at any optimization step $t$, it only maintains (and computes gradients with respect to) $Ld$ parameters, as opposed to $L|\mathcal{V}|$. This enables us to store much longer sequences in a GPU as compared to the storing $\tilde{\mathbf{y}}$. (b) This formulation provides a natural way to define hard rule-based constraints based on keywords or phrases (discussed in more detail in §6.2.3), and, finally (c) it yields a natural way to generate samples.

**Gradient based Sampling via Langevin Dynamics** The minimization in (6.7) can be very easily extended to a sampling procedure by modifying the gradient descent in (6.7) to Langevin Dynamics (Gelfand and Mitter, 1991; Welling and Teh, 2011),

$$\tilde{\mathbf{e}}^t = \text{Proj}_{\mathbf{E}}(\tilde{\mathbf{e}}^{t-1} - \eta \nabla_{\tilde{\mathbf{e}}} \mathcal{E}(\tilde{\mathbf{e}}^{t-1}) + \sqrt{2\eta\beta} z^t)$$

Langevin Dynamics provides a Monte Carlo Markov Chain (MCMC) method to sample from a distribution using only the gradient of its logarithm. That is, if we define a distribution as $Q(\mathbf{y}) \propto \exp(-\mathcal{E}(\mathbf{y}))$, its logarithm leads to the update specified above.[14] This method is often used for non-convex optimization for training neural networks (Welling and Teh, 2011) due to its ability to escape local minima due to added noise and converge towards the global minima. In this work, we adapt it for inference (Song and Ermon, 2019).

Intuitively, by adding noise at every gradient step, this procedure intends to find outputs $\mathbf{y}$ that do not exactly minimize $\mathcal{E}$ but remain in the vicinity of the minima. In other words, it finds outputs which admit high probability under the distribution $Q(\mathbf{y})$. This process begins with an exploration phase which is controlled by $\beta$. With a high value of $\beta$, the noise term is large leading to big updates. By gradual annealing such that $\beta \to 0$, as $t \to \infty$, this process converges to a sample from $Q(\mathbf{y})$.[15]

**Energy as a function of embeddings** We represent the energy function again as a Lagrangian as described in Equation 6.3 and perform gradient descent on the tokens (represented as embeddings) and gradient ascent on the multipliers. Again the intuition remains the same, if a constraint is not satisfied, the term $(\epsilon_i - f_i(\cdot))$

[13] Prior work (Belinkov and Glass, 2019) has shown that neural-network based models are not robust to change in input embedding space where changing the input vector to anything other than vectors from the embedding table can fool the model. We observed this phenomenon in our preliminary experiments where, without any projection, most low-energy solutions were found to be adversarial examples where they had high probability but were garbled text.

[14] The normalization term in $Q(\mathbf{y})$ vanishes as its gradient with respect to $\mathbf{y}$ is 0.

[15] More details of the implementation of annealing schedule can be found in §6.1.1. A similar noise can also be applied directly to the soft-token representations in §6.1 as explored in Qin et al. (2022). However, as we discuss in §6.2.4, our formulation with its smaller parameter size allows generating longer sequences. In addition, considering logits as soft-representations (followed by softmax) has shown to result in slow mixing, that is, it takes much longer to converge as empirically shown in Hoang et al. (2017) and also observed in Qin et al. (2022). On the other hand, considering the simplex itself (Kumar et al., 2021b; Hoang et al., 2017) as soft-representations is not compatible with Gaussian noise and can lead to undesirable behavior (Patterson and Teh, 2013).

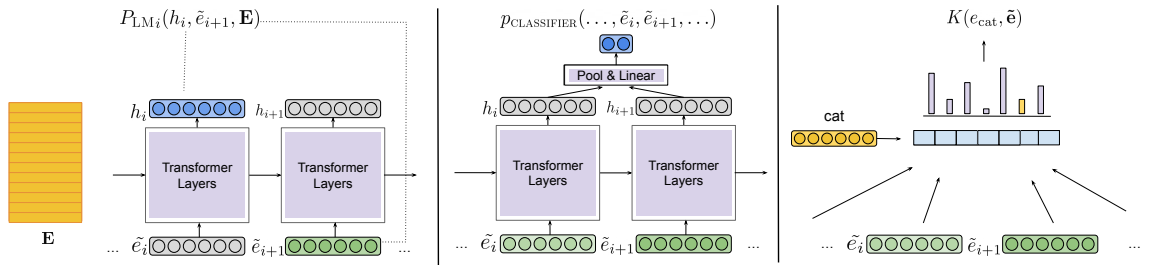



Figure 6.3: Different kinds of functions can be incorporated into MUCOCO defined on a shared embedding table $\mathbf{E}$. (Left) Language Modeling objective defines a per-token loss directly on the sequence of embeddings. For every token this loss provides gradients to update $\tilde{e}_i$ via backpropagation through the transformer layers and directly to $\tilde{e}_{i+1}$ through the negative loss likelihood loss as computed in §6.2. This is used as a primary objective for the underlying LM and can also be used for classification as discussed in §6.2.2 (Center) Classification objective defined on probability of the desired label. The classifier gets the token embeddings $\tilde{\mathbf{e}}$ directly as input and updates the embedding using gradients obtained via backpropagation from the transformer layers (Right) Lexical loss defined on the embeddings directly (without the use of additional models) to include desired keywords or phrases in the output sequence (§6.2.3). In practice any combination of these constraints can be used.

would be negative, and $\lambda_i$ would keep increasing making $\mathcal{E}$ high. On the other hand, if all the constraints are satisfied these values gradually decrease to 0 making $\mathcal{E}(\mathbf{y}) = -\log P(\mathbf{y})$ making the final output a sample from the desired distribution $P$. Again, we implement a damped version of this process to improve stability.

Further, performing gradient updates with respect to $\tilde{\mathbf{e}}$ requires that all objectives be defined as functions of $\tilde{\mathbf{e}}$, not $\mathbf{y}$. Also, $f_1(\mathbf{y}), \ldots, f_C(\mathbf{y})$ must share the same input embedding table (as that of $P$). We discuss in §6.2.1 how this can achieved for different kinds of constraint functions $f_i$. First, we describe how to compute the primary objective $-\log P(\mathbf{y}|\mathbf{x};\theta)$ and its gradients with respect to $\tilde{\mathbf{e}}$. In typical LMs, this objective is factorized as $\log P(\mathbf{y}|\mathbf{x}) = \sum_{n=0}^{L-1} \log P(y_{n+1}|y_{1:n}, \mathbf{x})$. For each decoding step $n+1$: the model takes as input $y_n$, which is converted to $e_n$ via an embedding table lookup. Passed through the network layers, it is converted to a hidden vector $h_n$. Since the input and output embedding tables in most modern LMs are shared (Radford et al., 2019a; Raffel et al., 2020; Lewis et al., 2020; Brown et al., 2020a),[16] the softmax probability is computed as,

$$P(y_{n+1}|y_{1:n}, \mathbf{x}) = \frac{\exp(h_n^T e_{n+1} + b_{n+1})}{\sum_{j=1}^{|\mathcal{V}|} \exp(h_n^T e_j + b_j)} \tag{6.8}$$

where $b_n$ are optional bias terms. By replacing $e_{n+1}$ with $\tilde{e}_{n+1}$, we convert the above probability to $P(\tilde{e}_{n+1}|\tilde{e}_{1:n}, \mathbf{x})$. For each position $n+1$, $\tilde{e}_{n+1}$ receives gradients, (a) directly from $-\log P$ function and (b) through $h_{n+1}$ via back-propagation through the network layers (See figure 6.3 (left)).

We call this algorithm MUCOLA for text generation with **Mu**ltiple **Co**nstraints via **La**ngevin Dynamics. The final decoding algorithm we used in our experiments is described in algorithm 3.

[16] Even if the embedding tables are not shared, this loss may be computed and optimized using vectors from the output embedding table as parameters without any significant loss in performance.

**Algorithm 3:** MUCOCO: detailed decoding algorithm

**Input:** input sequence $\mathbf{x}$, output length $L$, base LM, attribute functions $f_i$ and their respective thresholds $\epsilon_i$, step sizes $\eta$, $\eta_{max}$ (and schedule), $\eta_\lambda$, initial noise variance $\beta_{\text{init}}$ (and schedule);
**Result:** output sequence $\mathbf{y}$
For all $n \in \{1, \ldots, L\}$, initialize $\tilde{e}_n^{(0)}$;
For all $i \in \{1, \ldots u\}$, initialize $\lambda_i^{(0)}$ as 0;
Initialize $\beta^{(0)}$ as $\beta_{\text{init}}$;
Initialize $\eta^{(0)}$ as $\eta$;
**for** $t = 1, \ldots,$ MAXSTEPS **do**
  `// forward pass`
  compute the energy function $\mathcal{E}$ (see §6.2);
  `// backward pass`
  for all $n, i$, compute $\nabla_{\tilde{e}_n}^{(t-1)} = \frac{\partial \mathcal{E}}{\partial \tilde{\mathbf{e}}_n}$, $\nabla_{\lambda_i}^{(t-1)} = \frac{\partial \mathcal{E}}{\partial \lambda_i}$;
  `// Update the parameters`
  Sample $z^{(t-1)} \sim \mathcal{N}(0, I_d)$;
  Update $\tilde{\mathbf{e}}_{\mathbf{y}}^t = \text{Proj}_{\mathbf{E}}(\tilde{\mathbf{e}}_{\mathbf{y}}^{(t-1)} - \eta \nabla_{\tilde{\mathbf{e}}_{\mathbf{y}}}^{(t-1)} \mathcal{E} + \sqrt{2\eta^{(t-1)}\beta} z^{(t-1)})$;
  Update $\lambda_i^t = \max(0, \lambda_i^{t-1} + \eta_2 \nabla_{\lambda_i}^{(t-1)} \mathcal{E})$;
  update $\beta^{(t)}, \eta^{(t)}$ following the threshold update schedule.
**end**
Convert $\tilde{\mathbf{e}}^{(t)}$ to discete tokens $\hat{\mathbf{y}}^{(t)}$ by nearest neighbor search.;
**return** $\arg\min_t \{-\log P(\hat{\mathbf{y}}^{(t)}|\mathbf{x}) : \forall i, f_i(\tilde{\mathbf{y}}^{(t)}|[\mathbf{x}]) \leq \epsilon_i\}$;

### 6.2.1 Experimental Setup

We evaluate MUCOLA on four constrained generation tasks [17]. These tasks are selected based on defining different kinds of constraints for which prior work designed specialized training or decoding mechanisms which cannot be generalized beyond those tasks or language models. The main contribution of MUCOLA is generating diverse samples which conform to the language model $P$ as well as can satisfy user defined arbitrary combination of constraints for which fine-tuning is generally infeasible and tuning weights of each constraint is cumbersome. For a pre-defined sentence length $L$, we initialize the token representation for each step $\tilde{e}_1, \ldots, \tilde{e}_L$ using token embeddings randomly sampled from the target vocabulary $\mathcal{V}$.[18] For all our experiments, we run the Langevin Dynamics simulation for a maximum of 250 iterations unless specified otherwise.

**Noise Schedule** The amount of noise in each update is controlled by $\beta$ which represents the variance of the noise term. We initialize $\beta$ with 5.0 and decrease it to 0.05 in a geometric progression for 100 steps after which we keep it constant at 0.05 for the remaining 150 steps. The range of $\beta$ is guided by best practices in Song and Ermon (2019) prescribing the initial variance to be close to the maximum distance between any two vectors in the input space and the minimum value being close to 0. This schedule allows for sufficient exploration in the beginning helping in diversity, while, leaving enough iterations for optimizing the final output into a fluent sequence.

[17] Similar to MUCOCO, MUCOLA is also interesting to evaluate in a unconstrained setting by simply sampling from the LM. In initial explorations, we found this method to replicate results in terms of perplexity on ancestral sampling with LMs including its issues such as repetitions. Future work may explore constraints to mitigate such issues.

[18] We also tried other initialization strategies like initializing with zeros, or outputs of nucleus sampling or greedy decoding but did not find it to have any significant effect on the final output

**Step Size and Selection Criterion** The step-size $\eta$ in projected gradient descent depends on the geometry of the embedding space of the underlying language model. Since we project back the update at every step to $\mathbf{E}$, if the update term is not big enough in the projected gradient update, the sequence at step $t+1$ would remain the same. This observation provides a very simple criterion for early stopping and selecting the best output out of all iterations. When the additive noise is small (near the end of optimization), the update term can be small due to following factors: (a) $\eta$ is small, (b) the gradient $\nabla \mathcal{E}_{\tilde{\mathbf{e}}}$ is small which implies the output sequence has "converged". Hence, we define a schedule on the step-size as follows: we start with a step-size $\eta$, and update the outputs using Langevin Dynamics until the sequence stops updating, i.e., the update value becomes too small (and satisfies all constraints). Now, to make sure that this is a convergence point and not a result of the step size being too small, we update the step size linearly to $\eta_{\text{max}}$ in $s$ steps[19]. If the sequence does not update in $s$ steps, we stop early and predict the output. Otherwise, the process continues. If it does not stop early at the end of maximum number of steps, we predict the output with the highest likelihood which repeated at least 5 times. In the event, no such repetition is observed, we deem the optimization as "failed" and restart. If the restarts also fail, we just predict the autoregressive output (which in our experiments is obtained with nucleus sampling with $p = 0.96$). This fallback mechanism ensures that the output, irrespective of the constraint satisfaction is always a sample of $P$ while preventing generating half-baked outputs.

**Multipliers Update Schedule** We initialize each of the multipliers $\lambda_i$ with 0, update the multipliers via gradient ascent every 20 steps using the step-size 1.0. In addition, if the sequence stops updating at a certain iteration (as described above) and $i$-th constraint is not satisfied, we update $\lambda_i$ at every iteration till the sequence starts updating again. This schedule prevents fluctuation in the multiplier values when the noise is high in the early iterations and the sequence has not converged to anything fluent while still allowing updates when required (Platt and Barr, 1988; Paria et al., 2020).

### 6.2.2 Text Generation with Soft Constraints

First, we evaluate MUCOLA with real valued constraint functions defined via auxiliary models such as classifiers or LMs. Given an LM GPT2-Large (Radford et al., 2019a), and a prompt $\mathbf{x}$, we generate continuations $\mathbf{y}$. We conduct experiments with: toxicity avoidance and sentiment control. Each of the tasks define a binary constraint. Let the desired label be denoted by $\text{LABEL}_1$, and other one with $\text{LABEL}_0$ ($\text{LABEL}_1$ is non-toxic in toxicity avoidance and positive in sentiment control). For both setups, we assume availability of corpora to train the constraint functions.[20]

**Baselines** In addition to decoding without any constraints (which we simply call GPT2), we consider the following baselines which decode from left-to-right:

- **Domain Adaptive Pretraining (DAPT)** (Gururangan et al., 2020) proposes to finetune the LM $P$ on a corpus of desired constraint and sample directly from finetuned version.
- **FUDGE** (Yang and Klein, 2021) uses a "future-aware" constraint classifier to modify output token probabilities at every decoding step to steer the generation to promote constraint satisfaction. This classifier is trained to predict the ground truth label for every prefix of the training corpus.

[19] $s$ is empirically defined as 40 in all our experiments.
[20] This setup can be easily extended to $n$-class setups by defining $n-1$ constraints as $p_0 > p_1, \ldots, p_0 > p_{n-1}$

- **GeDi** (Krause et al., 2020b) uses a class-conditioned LM to modify output token probabilities at each step via Bayes' rule.

- **DExperts** (Liu et al., 2021) proposes to replace the class-conditioned LM with two auxiliary language models (one expert and one anti-expert) to modify the output logits at every step. These LMs are trained using same setup as the baseline **DAPT** but instead of sampling from them directly, it uses them to steer the base LMs outputs. For each of the baselines, we use recommended hyperparameters to generate samples.

Since we decode by computing gradients over token embeddings, it requires that all constraint models share the same embedding table $\mathbf{E}$ as that of the underlying language model $P$. Since any typical text based model involves an embedding table, we can train a constraint using such a model by simply initializing its embedding table with $\mathbf{E}$. In principle, this initialization allows using any off-the-shelf pretrained model as a constraint function by finetuning it on appropriate data. Each of the baselines we described above can be adopted as constraint functions for MUCOLA as follows:

- **Discriminative Classifiers** We train a binary classifier $p_{\text{LABEL}}(\mathbf{y})$, which predicts the probability of the desired attribute given the output sequence $\mathbf{y}$ by finetuning roberta-base with GPT2-Large embeddings.To decode with MUCOLA, we formulate this constraint as $p_{\text{LABEL}_1} > \epsilon$ (We define specific $\epsilon$ values in §6.2.2 and §6.2.2 respectively). To improve its gradient profile, we use the constraint in log space. We call this setup **MUCOLA-DISC**.

- **Generative Classifiers** Prior work has shown that discriminative classifiers can be fragile to domain shift or adversarial examples (Yogatama et al., 2017b; Krause et al., 2020a). Hence, we also consider a second class of *generative* classifiers trained as class conditional LMs that model $p(\cdot|\text{LABEL})$. Intuitively, they are required to explain every word in the input, potentially amplifying the class signal and improving robustness (Min et al., 2021). We define them in three ways by finetuning GPT2-Large: (1) following GEDI (**MUCOLA-GeDi**), (2) following DExperts, (we train two separate LMs; **MUCOLA-DExperts**). And finally, (3) motivated by recent work on prompt-based classification, we define a class-conditional LM without finetuning the model as $P(\mathbf{x}, \mathbf{y}|\text{verbalize}(\text{LABEL}))$ where verbalize$(\cdot)$ is function that converts the label to a natural language string (**MUCOLA-prompt**). Note that for all three setups, the embedding table is frozen. We decode via MUCOLA with the constraint $p(\mathbf{x}, \mathbf{y}|\text{LABEL}_1)) > p(\mathbf{x}, \mathbf{y}|\text{LABEL}_0)$ (again, realized in log-space)[21].

**Evaluation** In both experiments, we evaluate the generated samples along three dimension, (1) **Constraint Satisfaction** measured using external evaluators, (2) **Fluency**, measured by mean perplexity of the continuations measured using GPT2-XL. Since the objective is to generate samples from the LM, we rank different methods not by their absolute perplexity, but its difference from the perplexity of unconstrained text. Additionally, we also report a grammaticality score: the fraction of outputs predicted by a classifier trained on CoLA (Warstadt et al., 2019) as fluent. (3) **Diversity**, measured by computing the mean number of distinct n-grams in each set of samples, normalized by the length of text (Li et al., 2016). We report this for $n = 1, 2, 3$ following prior work. Since all the automatic metrics are model based and can be biased, we also perform human evaluation in an A/B testing setup with the best performing baseline (DExperts in our case). For each

[21] Note that all constraints we describe can be easily extended to $n$-class set (with say 0 as the desired label) by defining $n - 1$ constraints as $p_0 > p_1, \ldots, p_0 > p_{n-1}$

| Approach | Toxicity | | Fluency | | Diversity | | |
|---|---|---|---|---|---|---|---|
| | Avg. Max. Toxicity | Toxicity Prob. | Perplexity | CoLa Accuracy | Dist-1 | Dist-2 | Dist-3 |
| GPT-2 | 0.527 | 0.520 | 25.45 | 88.3 | 0.58 | 0.85 | 0.85 |
| DAPT | 0.428 | 0.360 | 31.21 | **91.2** | 0.57 | 0.84 | 0.84 |
| FUDGE | 0.437 | 0.371 | 12.97 | 88.5 | 0.47 | 0.78 | 0.82 |
| GEDI | 0.363 | 0.217 | 60.03 | 85.5 | 0.62 | 0.84 | 0.83 |
| DEXPERTS | **0.302** | 0.118 | 38.20 | 89.8 | 0.56 | 0.82 | 0.83 |
| MUCOLA | **0.308** | **0.088** | **29.92** | 88.2 | 0.55 | 0.82 | 0.83 |

Table 6.9: Results for toxicity avoidance (§6.2.2). We evaluate on three axes: (1) Toxicity–Avg. Max. Toxicity and Toxicity Prob.: lower the better. (2) Fluency–GPT2-XL Perplexity: the closer the value to unconstrained outputs (GPT2: 38.6), the better; CoLa accuracy: higher the better, and (3) Diversity (Dist-1,2,3): higher the better. The best values in each column are highlighted in **bold**. While our method improves or performs on par with baselines on toxicity metrics, we obtain substantial improvements on perplexity.

sample, we ask 3 annotators to compare and rank candidates from our approach and the baseline on constraint satisfaction, topicality and fluency.

### Toxicity Avoidance

Prior work have shown that large pre-trained LMs are at risk of producing toxic content even when given innocuous prompts (Sheng et al., 2019; Gehman et al., 2020). In this experiment, given a neutral prompt, we generate non-toxic continuations using MUCOLA. We only consider the setup MUCOLA-DISC here, with a classifier $p_{\text{TOXIC}}$, trained on a dataset of human-annotated comments labeled as toxic or non-toxic. We decode with the constraint $p_{\text{TOXIC}} < 0.01$. We train the constraint function by finetuning roberta-base (Liu et al., 2019) with a binary classification head using a dataset of human-annotated comments from the Jigsaw Unintended Bias In Toxicity Classification Kaggle Challenge. The dataset has $\sim 160K$ toxic comments and $\sim 1.4M$ non-toxic comments. We first balance this dataset by subsampling $160K$ examples from the non-toxic class. We replace the embedding table of roberta-base with that of the underlying LM (GPT2-Large in our case). To address the dimension mismatch of the two embedding tables, during finetuning, we also learn a linear projection matrix which transforms base LM embedding to a smaller dimension of roberta-base. We keep base LM embedding frozen during this finetuning. We use a learning rate of $1e-5$ and train for 3 epochs with an effective batch size of 64. We choose a checkpoint with an accuracy of $\sim 93\%$ on a heldout development set.

We follow the evaluation setup defined in Liu et al. (2021) and use a test set of 10K nontoxic prompts (Gehman et al., 2020) where without any constraints, the user might receive harmful output from the LM. For each prompt, we generate 25 samples for length 20 tokens each. We measure constraint satisfaction using the toxicity score from Perspective API. Following prior work (Gehman et al., 2020; Liu et al., 2021), we report the maximum toxicity score over 25 samples per prompt averaged over the number of prompts, and the empirical probability of generating a continuation with toxicity $> 0.5$ at least once over the 25 generations.

As shown in Table 6.9, MUCOLA outperforms or matches all baselines on toxicity, including a strong baseline DEXPERTS which is specifically designed for binary constraints. In addition, our method is closest in perplexity to unconstrained generation, while maintaining grammaticality as well as diversity of baseline methods[22]. We attribute this improvement to the fact that after the constraints are satisfied, the energy function

[22]While FUDGE obtains the lowest absolute perplexity, prior work (Holtzman et al., 2020) has shown that very low perplexity is not an

| Approach | Setting | % Positive Sentiment | | | Fluency | | Diversity | | |
|---|---|---|---|---|---|---|---|---|---|
| | | C1 | C2 | C3 | Perplexity | CoLa | Dist-1 | Dist-2 | Dist-3 |
| GPT-2 | - | 46.7 | 47.7 | 61.3 | 38.6 | 78.7 | 0.64 | 0.90 | 0.88 |
| DAPT | SST-2 | 73.6 | 70.0 | 78.3 | 76.9 | 70.7 | 0.64 | 0.89 | 0.86 |
| FUDGE | SST-2 | 67.6 | 63.0 | 79.3 | 10.3 | **94.0** | 0.51 | 0.80 | 0.84 |
| GEDI | SST-2 | 99.0 | 96.3 | 99.7 | 268.7 | 54.0 | 0.69 | 0.87 | 0.84 |
| DEXPERTS | SST-2 | 91.2 | 83.4 | 95.4 | 55.37 | 81.6 | 0.61 | 0.89 | 0.87 |
| MUCOLA-DISC | SST-2 | 84.6 | 77.5 | 88.0 | 27.9 | 80.8 | 0.50 | 0.81 | 0.82 |
| MUCOLA-DISC | Yelp | 83.0 | 83.6 | 83.0 | **32.2** | 76.0 | 0.50 | 0.75 | 0.80 |
| MUCOLA-TWO-DISC | Yelp, SST-2 | **93.7** | **91.0** | **96.0** | 28.9 | 76.7 | 0.53 | 0.77 | 0.74 |
| MUCOLA-PROMPT | - | 87.3 | 91.0 | 93.0 | 53.0 | 77.2 | 0.54 | 0.82 | 0.80 |

Table 6.10: Results for Sentiment Controlled Generation for outputs of length 20. We evaluate on three axes: (1) % Positive Sentiment: higher the better. We use three external classifiers for this evaluation, C1 trained on SST2 data, C2 trained on Yelp data, and C3 trained on 15 polarity datasets; (2) Fluency–GPT2-XL perplexity, closer the value to unconstrained outputs (GPT2: 38.6), the better; CoLa accuracy: higher the better, and (3) Diversity (Dist-1,2,3): higher the better. The best values in each column are highlighted in **bold**.

in MUCOLA reduces to $-\log P(\mathbf{y})$, the original function we intend to sample from, whereas in the baselines, the underlying probability distribution (or the energy function) is modified to achieve control. We further conduct human evaluation to strengthen these results. For human evaluation, we follow an A/B testing framework and compare MUCOCO and DExperts. We sample 200 prompts from the test set and consider 2 generations per prompt. We ask each annotator to rank the outputs from the two approaches on (1) toxicity if one output is more or less toxic than the other, or if both are equally toxic/non-toxic, (2) topicality: is the generation coherent with the prompt and follows the same general topic, and (3) fluency: if the outputs have any grammatical mistakes. We collect 3 annotations per pair. We find that in terms of toxicity, both models perform similarly with an average 8.5% annotations preferring MUCOCO's outputs compared to 9.5% for DExperts (rest are equally ranked). On topicality, 22.5% of annotations prefer MUCOCO's outputs while 19% prefer Dexperts (rest are equally ranked). On fluency, both models perform similarly with 22.5% and 23% in each method's favor and rest equally ranked. Overall, human evaluation reveals that generations by MUCOLA match DExperts on toxicity and fluency while being more topical.

### Sentiment Controlled Generation

Given a prompt $\mathbf{x}$, the goal of this task is to generate continuations $\mathbf{y}$ using an LM with a positive sentiment/polarity. To understand the effect of sources of training data, we train two versions of each constraint function described above on two datasets: SST-2 corpus (Socher et al., 2013) containing $\sim$4K examples in Movie reviews for each class; and Yelp polarity corpus containing $\sim$280K examples for each class containing a mixed domain of reviews. We also consider an additional setup where we use two constraints using both versions of **MUCOLA-DISC**, which we call **MUCOLA-TWO-DISC**.

For discriminative classifiers, we again finetune roberta-base using the same setup and hyperparameters as the toxicity classifier. Our best model obtains an accuracy of $\sim$92% on the SST-2 test set and $\sim$98% on the Yelp test set. To train the generative classifiers, we finetune GPT2-Large (and do not need to substitute any embedding tables) keeping the embedding table frozen. We use the loss $-\log p_{\text{gen}}(\text{label}|x)$ for each training instance where $p_{\text{gen}}(\text{label} = 0|\text{text}) = p_{\text{LM}}(\text{text}|\text{label} = 0)/(p_{\text{LM}}(\text{text}|\text{label} = 0) + p_{\text{LM}}(\text{text}|\text{label} = 1))$. This is due to Bayes' rule ($p(\text{label})$ vanishes as we set it to 0.5 for balanced datasets). Here $p_{\text{LM}}(\text{text}|\text{label})$ is

indicator of higher quality but of repetitions and usage of only high frequency tokens.

obtained using the language model by computing the probability of the text conditioned on the input token "positive" for the positive label and "negative" otherwise. We again follow the same training hyperparameters for this setup. On SST-2 test set, we obtain an accuracy of $\sim$95% and on Yelp, we obtain an accuracy of $\sim$98%.

We use a dataset of 15 prompts from Dathathri et al. (2020) and generate 20 samples per prompt of length 12, 20, and 50. To evaluate constraint satisfaction, we measure positive sentiment accuracy of the output using three external classifiers to account for domain differences in their training data , (a) **C1**: distilbert (Sanh et al., 2019) finetuned on SST-2 data, used in (Liu et al., 2021), (b) **C2**: bert-base (Devlin et al., 2019) finetuned on Yelp Polarity corpus used in Mireshghallah et al. (2022a), and (c) **C3**: SieBERT (Heitmann et al., 2020) finetuned on 15 different polarity datasets.

We report a subset of the results of this experiment in table 6.10 for outputs of length 20 for clarity (the full set of results can be found in tables 6.11, 6.12, and 6.13). We observe a significant variance in sentiment control accuracies (C1, C2 and C3) where constraints trained on SST-2 perform worse on the evaluator trained on Yelp (C2) and vice versa for all methods. The third evaluator (C3) trained on a much larger training set can be considered more reliable. Overall, we find that MUCOLA in all settings obtains perplexity values closer to unconstrained outputs (GPT2) whereas most baselines achieve control at the cost of perplexity. Surprisingly, constraints trained on Yelp perform poorly compared to those trained on SST2 despite the former being a larger dataset.

For outputs of lengths 12 and 20, MUCOCO-TWO-DISC finds a good balance of control and fluency and outperforms all other baselines on sentiment accuracy while maintaining good perplexity (except GEDI which performs poorly on perplexity as well as CoLa accuracy). This improvement however comes with a slight decline in diversity metrics which we argue is a fair price to pay for constraint satisfaction compared to fluency. Similar to toxicity avoidance, a small scale study on human evaluation reveals MUCOLA to be more topical than the best baseline DExperts. Finally, using a prompt-based constraint also performs strongly despite not trained at all. Future work may look into training a prompt-based classifier to improve this performance. For human evaluation, we follow an A/B testing framework and compare MUCOLA and DExperts (for outputs of length 20). We consider all 15 prompts from the test set and consider 2 generations per prompt. We ask each annotator to rank the outputs from the two approaches on (1) positivity if one output is positive and the other is not, or if both are positive/not-positive, (2) topicality: is the generation coherent with the prompt and follows the same general topic, and (3) fluency: if the outputs have any grammatical mistakes. We collect 3 annotations per pair. We find that in terms of positivity, on an average 23.3% annotations prefer MUCOLA's outputs compared to 16.7% for DExperts (rest are equally ranked). On topicality, 26.7% of annotations prefer MUCOLA's outputs while 13.3% prefer Dexperts (rest are equally ranked). On fluency, MUCOLA slightly underperforms with 7.8% and 10% in each method's favor and rest equally ranked.

For outputs of length 50, we observe a slight drop in MUCOLA's performance. On closer inspection (table 6.22), we find a trend of degenerate repetitions at the end of many sequences. Prior work (Holtzman et al., 2020) has shown that large LMs often assign unusually high probabilities to repeating sequences, especially with increasing lengths and since our method is designed to sample high probability outputs, such behavior is expected. Future work may explore constraints designed to discourage this behavior (Welleck et al., 2020; Meister et al., 2022).

| Approach | Setting | % Positive Sentiment (↓) | | | Fluency | | Diversity | | |
|---|---|---|---|---|---|---|---|---|---|
| | | C1 | C2 | C3 | Perplexity | CoLa Accuracy | Dist-1 | Dist-2 | Dist-3 |
| GPT-2 | - | 49.0 | 45.0 | 62.0 | 54.9 | 68.7 | 0.66 | 0.87 | 0.81 |
| DAPT | SST-2 | 71.3 | 66.7 | 75.0 | 98.0 | 64.0 | 0.64 | 0.85 | 0.79 |
| DAPT | Yelp | 64.0 | 71.3 | 79.7 | 146.6 | 58.0 | 0.60 | 0.84 | 0.80 |
| FUDGE | SST-2 | 71.7 | 70.0 | 79.0 | 11.4 | 82.7 | 0.53 | 0.76 | 0.77 |
| FUDGE | Yelp | 71.7 | 73.7 | 84.7 | 11.8 | 85.7 | 0.53 | 0.76 | 0.77 |
| MUCOCO-DISC | SST-2 | 90.0 | 81.7 | 93.3 | 28.8 | 67.3 | 0.52 | 0.73 | 0.74 |
| MUCOCO-DISC | Yelp | 88.3 | 87.0 | 91.7 | 32.9 | 64.3 | 0.52 | 0.74 | 0.75 |
| MUCOCO-TWO-DISC | Yelp, SST2 | 94.0 | 91.3 | 94.7 | 29.4 | 55.0 | 0.46 | 0.68 | 0.71 |
| GEDI | SST-2 | 99.7 | 91.0 | 99.3 | 625.7 | 54.3 | 0.65 | 0.76 | 0.71 |
| GEDI | Yelp | 82.0 | 90.0 | 89.0 | 444.9 | 40.0 | 0.71 | 0.78 | 0.66 |
| MUCOCO-GEN | SST-2 | 91.3 | 88.3 | 97.0 | 57.2 | 68.0 | 0.50 | 0.69 | 0.70 |
| MUCOCO-GEN | Yelp | 86.3 | 89.7 | 91.7 | 53.0 | 67.7 | 0.50 | 0.70 | 0.70 |
| MUCOCO-PROMPT | - | 89.0 | 88.7 | 94.7 | 43.7 | 66.7 | 0.49 | 0.72 | 0.73 |
| DEXPERTS | SST-2 | 93.1 | 86.9 | 94.9 | 75.2 | 71.5 | 0.63 | 0.85 | 0.81 |
| DEXPERTS | Yelp | 80.3 | 88.5 | 88.8 | 116.3 | 67.5 | 0.67 | 0.84 | 0.79 |
| MUCOCO-DEXPERTS | SST-2 | 93.0 | 88.0 | 94.0 | 41.4 | 66.3 | 0.47 | 0.71 | 0.73 |
| MUCOCO-DEXPERTS | Yelp | 74.3 | 74.0 | 83.3 | 72.5 | 66.0 | 0.52 | 0.73 | 0.74 |

Table 6.11: Positive sentiment control results on outputs of length 12. For each baseline (FUDGE, GEDI and DEXPERTS), we convert their respective constraints to a classifier (generative or discriminative; see §6.2.2). For FUDGE and GEDI, we show improvements on both control (% positive sentiment) and fluency (Perplexity) without any model specific changes. This improvement is consistent on models trained on both datasets (SST-2 and Yelp). DEXPERTS outperforms all baselines here including our method.

### 6.2.3 Decoding with Hard Constraints

In the previous two tasks, we explored how MUCOLA can be applied to soft constraints, defined via real-valued functions like probabilities of classifiers or language models which can be used to generate text with desired categorical variations. Now, we consider a ruled-based constraint that a specific word or phrase *must* appear in the generated text. Existing autoregressive solutions to this task have explored various strategies either based on explicitly modifying probabilities to up-weight desired words (Pascual et al., 2021), or search-based strategies based on beam-search (Lu et al., 2021b). We define a differentiable distance function $d(w, \tilde{\mathbf{e}})$ which measures the overlap between a desired word ($w$) and the output token embeddings $\tilde{\mathbf{e}}$ (we use the notation $w$ to refer to as the word itself and its index in the vocabulary interchangeably). We then propose a simple criterion to define a threshold $\epsilon$ that guarantees that if $d(w, \tilde{\mathbf{e}}) < \epsilon$, then $w$'s embedding appears in $\tilde{\mathbf{e}}$ (and by extension $w$ appears in $\mathbf{y}$). Taking inspiration from Liu et al. (2022); Qin et al. (2022), this function is computed in three steps. First, we convert each $\tilde{e}_n$ to a "probability" over the vocabulary as,

$$\pi_n = \text{softmax}(-\|\tilde{e}_n - e_1\|_2^2, \ldots, -\|\tilde{e}_n - e_{|\mathcal{V}|}\|_2^2)$$

where $\{e_1, \ldots, e_{|\mathcal{V}|}\}$ are entries in the embedding table $\mathbf{E}$. Since each $\tilde{e}_n$ itself also corresponds to a vector in $\mathbf{E}$, if $n$-th token in the sequence is $w$, then, $\|\tilde{e}_n - e_w\|_2^2$ would be 0 leading to $\pi_{n,w} = \max_j \pi_{n,j}$. That is, maximizing $g_n = \log \pi_{n,w}$ with respect to $\tilde{e}_n$ would nudge it towards the $e_w$. Since, we don't know which index we want $w$ to appear at in advance, we (soft) sample it using $\pi_{n,w}$ as weights. This brings us to the second step, as we define, $q = \text{GUMBEL-SOFTMAX}(-g_1/\tau, \ldots, -g_N/\tau)$ where $\tau$ is the temperature. We use hard sampling here to ensure $q$ is one-hot. Finally, we define the constraint function as, $d(w, \tilde{\mathbf{e}}) = \sum_{i=n}^{N} -q_n g_n$. Intuitively, this function aims to generate the word $w$ wherever it already has a high chance of getting generated

| Approach | Setting | % Positive Sentiment (↑) | | | Fluency | | Diversity | | |
|---|---|---|---|---|---|---|---|---|---|
| | | C1 | C2 | C3 | Perplexity | CoLa Accuracy | Dist-1 | Dist-2 | Dist-3 |
| GPT-2 | - | 46.7 | 47.7 | 61.3 | 38.6 | 78.7 | 0.64 | 0.90 | 0.88 |
| DAPT | SST-2 | 73.6 | 70.0 | 78.3 | 76.9 | 70.7 | 0.64 | 0.89 | 0.86 |
| DAPT | Yelp | 65.0 | 75.0 | 80.7 | 86.6 | 69.7 | 0.59 | 0.88 | 0.87 |
| FUDGE | SST-2 | 67.6 | 63.0 | 79.3 | 10.3 | 94.0 | 0.51 | 0.80 | 0.84 |
| FUDGE | Yelp | 71.0 | 70.0 | 79.3 | 10.6 | 89.0 | 0.53 | 0.81 | 0.85 |
| MUCOCO-DISC | SST-2 | 84.6 | 77.5 | 88.0 | 27.9 | 80.8 | 0.50 | 0.81 | 0.82 |
| MUCOCO-DISC | Yelp | 83.0 | 83.6 | 83.0 | **32.2** | 76.0 | 0.50 | 0.75 | 0.80 |
| MUCOCO-TWO-DISC | Yelp, SST2 | **93.7** | **91.0** | **96.0** | 28.9 | 76.7 | 0.53 | 0.77 | 0.74 |
| GEDI | SST-2 | 99.0 | 96.3 | 99.7 | 268.7 | 54.0 | 0.69 | 0.87 | 0.84 |
| GEDI | Yelp | 84.0 | 95.7 | 91.0 | 208.3 | 44.0 | 0.76 | 0.87 | 0.81 |
| MUCOCO-GEN | SST-2 | 86.3 | 80.3 | 93.3 | 45.6 | 77.7 | 0.50 | 0.74 | 0.78 |
| MUCOCO-GEN | Yelp | 79.7 | 83.0 | 90.0 | 27.2 | 72.3 | 0.50 | 0.82 | 0.86 |
| MUCOCO-PROMPT | - | 87.3 | 91.0 | 93.0 | 53.0 | 77.2 | 0.54 | 0.82 | 0.80 |
| DEXPERTS | SST-2 | 91.2 | 83.4 | 95.4 | 55.37 | 81.6 | 0.61 | 0.89 | 0.87 |
| DEXPERTS | Yelp | 81.1 | 85.8 | 92.5 | 95.87 | 71.7 | 0.66 | 0.89 | 0.87 |
| MUCOCO-DEXPERTS | SST-2 | 89.3 | 83.7 | 93.7 | **32.2** | 79.7 | 0.51 | 0.78 | 0.80 |
| MUCOCO-DEXPERTS | Yelp | 78.0 | 75.7 | 83.3 | 34.1 | 68.3 | 0.52 | 0.77 | 0.81 |

Table 6.12: Positive sentiment control results on outputs of length 20. For each baseline (FUDGE, GEDI and DEXPERTS), we convert their respective constraints to a classifier (generative or discriminative; see §6.2.2). For FUDGE and GEDI, we show improvements on both control (%positive sentiment) and fluency (Perplexity) without any model specific changes. This improvement is consistent on models trained on both datasets (SST-2 and Yelp).

(measured via $\pi_{n,w}$'s). Stochasticity in this function allows for exploration. This function can be easily extended from words to phrases of length $l$, $w = (w_1, \ldots, w_l)$ by defining $g_n = \frac{1}{l}\sum_{u=1}^{l} -\log \pi_{w_u, n+u}$. This computation can be efficiently done on a GPU using a convolution operation (Liu et al., 2022).

Based on this definition, we define the keyword constraint for MUCOLA as $d(w, \tilde{\mathbf{e}}) \leq -\log \pi_{w,w} + \delta$, where $\delta$ is a small positive value (we set it as 0.1). $\pi_w$ is a slight abuse of notation to define a distribution similar to $\pi_n$ ($n$ refers in an index in sequence whereas $w$ refers to an index in $\mathcal{V}$). Note that the threshold for each keyword is different.[23]

Intuitively, if $w$ appears in the output at the $k$-th position, then $\pi_{k,w} = \pi_{w,w} = \max_j \pi_{k,j}$ with $q_k$ as 1. This reduces the distance function to $-\log \pi_{k,w}$ which is less than the defined threshold. Conversely, if $w$ does not appear in the output, for each $n$, $-\log \pi_{n,w}$ would be higher than $\log \pi_{w,w}$ and the constraint remains unsatisfied. This is due to an empirical observation we make in all embedding tables we use, that $\pi_{w,w} = \max_j \pi_{w,j} = \max_j \pi_{j,w}$. In other words, not only is the probability of a word under its own distribution $\pi_w$ the greater than probability of all other words (since the corresponding distance is 0), it is also larger than $w$'s probability under all other distributions defined for any word in the vocabulary. Under the assumption that minimum distance between any two distinct vectors in the table is greater than a small positive value, we conjecture this claim to be true for any embedding table.

**Tasks** We formally evaluate this setup on two tasks: (1) open-ended keyword constrained generation (with two datasets: COMMONGEN and ROC)), and (2) terminology constrained machine translation. We additionally show preliminary findings on a third task, entity guided summarization.

[23] While we do not experiment with it in this work, the constraint $K(w, \tilde{\mathbf{e}})$ can be easily extended to setup where at least one out of $n$ given words (for example different surface forms of the same root), $S = \{w_1, \ldots, w_p\}$ must appear in the output by defining a new constraint as $K(S, \tilde{\mathbf{e}}) = \max_{w_i \in S} K(w_i, \tilde{\mathbf{e}})$ or its soft version using the gumbel-softmax trick.

| Approach | Setting | % Positive Sentiment (↓) | | | Fluency | | Diversity | | |
|---|---|---|---|---|---|---|---|---|---|
| | | C1 | C2 | C3 | Perplexity | CoLa Accuracy | Dist-1 | Dist-2 | Dist-3 |
| GPT-2 | - | 47.7 | 44.3 | 61.3 | 36.3 | 78.3 | 0.59 | 0.92 | 0.94 |
| DAPT | SST-2 | 93.0 | 84.3 | 91.7 | 55.3 | 88.0 | 0.61 | 0.92 | 0.94 |
| DAPT | Yelp | 72.3 | 80.7 | 85.0 | 46.1 | 84.3 | 0.51 | 0.90 | 0.94 |
| FUDGE | SST-2 | 71.0 | 61.3 | 84.7 | 8.5 | 98.3 | 0.47 | 0.83 | 0.92 |
| FUDGE | Yelp | 72.3 | 68.0 | 80.3 | 8.3 | 99.0 | 0.47 | 0.83 | 0.92 |
| MUCOCO-DISC | SST-2 | 88.7 | 81.0 | 91.3 | 15.3 | 72.7 | 0.42 | 0.68 | 0.76 |
| MUCOCO-DISC | Yelp | 70.7 | 74.3 | 81.3 | 19.1 | 77.7 | 0.48 | 0.77 | 0.85 |
| MUCOCO-TWO-DISC | Yelp, SST2 | 94.0 | 91.3 | 94.7 | 29.4 | 75.0 | 0.57 | 0.78 | 0.79 |
| GEDI | SST-2 | 86.7 | 98.7 | 96.7 | 148.4 | 68.3 | 0.75 | 0.94 | 0.93 |
| GEDI | Yelp | 99.7 | 98.7 | 100.0 | 114.5 | 74.3 | 0.66 | 0.93 | 0.93 |
| MUCOCO-GEN | SST-2 | 85.0 | 76.3 | 91.0 | 22.5 | 63.7 | 0.44 | 0.71 | 0.78 |
| MUCOCO-GEN | Yelp | 77.7 | 80.7 | 88.3 | 23.4 | 65.0 | 0.43 | 0.69 | 0.76 |
| MUCOCO-PROMPT | - | 81.3 | 83.0 | 92.7 | 18.2 | 72.0 | 0.39 | 0.67 | 0.77 |
| DEXPERTS | SST-2 | 98.1 | 92.0 | 99.5 | 39.5 | 88.5 | 0.57 | 0.91 | 0.94 |
| DEXPERTS | Yelp | 87.2 | 91.7 | 94.9 | 54.0 | 77.3 | 0.62 | 0.92 | 0.93 |
| MUCOCO-DEXPERTS | SST-2 | 72.7 | 71.7 | 84.7 | 28.2 | 69.0 | 0.45 | 0.75 | 0.83 |
| MUCOCO-DEXPERTS | Yelp | 62.3 | 61.7 | 75.7 | 18.8 | 81.0 | 0.48 | 0.77 | 0.83 |

Table 6.13: Positive sentiment control results on outputs of length 50. For each baseline (FUDGE, GEDI and DEXPERTS), we convert their respective constraints to a classifier (generative or discriminative; see §6.2.2). For FUDGE and GEDI, we show improvements on both control (% positive sentiment) and fluency (Perplexity) without any model specific changes. This improvement is consistent on models trained on both datasets (SST-2 and Yelp).

| | Coverage | | Fluency | |
|---|---|---|---|---|
| | Count | Percent | Perplexity | Human |
| TSMH | 2.72 | 71.27 | 1545.15 | 1.72 |
| Neurologic | 3.30 | 91.00 | 28.61 | **2.53** |
| COLD | 4.24 | 94.5 | 54.98 | 2.07 |
| MUCOLA | **4.49** | **99.7** | **23.50** | 2.29 |

Table 6.14: Results of keyword constraint on COMMONGEN. We report (a) coverage as avg. count of desired keywords in the output and the fraction of the outputs containing all keywords (percent); and (b) GPT2-XL perplexity and avg. fluency score rated by humans.

**Open-Ended Keyword Guided Generation** Following prior work, we measure the performance on two axes, (1) **Coverage**, measured by (a) count average number of keywords appearing in the output; and (b) percent, measuring the fraction of outputs which contain all the desired keywords. (2) **Fluency**, as measured by GPT2-XL perplexity and human evaluation (for COMMONGEN), where on a sample of 200 outputs, we ask 3 annotators to rate each output on a 3-point likert scale. In COMMONGEN (Lin et al., 2020) given no prompt , the task is generate an output of maximum length 40 which contains a given set of four or five words. We use GPT2-XL as the underlying LM in this setup with COLD (Qin et al., 2022) as our main baseline. In addition, we report results on ROC (Pascual et al., 2021) task where given 5 keywords, the goal is generate a sequence of max length 90 containing those terms. For both datasets, for set of keywords, we generate samples of length 10, 20, and 40 (with 3 restarts for each) and after all iterations are complete, we continue generating more tokens autoregressively until a maximum of 40 (90 in case of ROC) tokens are generated or end of sequence token is generated. Finally, we evaluate on one output which satisfies the constraints and has the lowest perplexity according to the LM. We compare MUCOLA with the best reported results

| | Coverage (%) | Fluency (PPL) | Repetition Rate |
|---|---|---|---|
| Plan-and-Write | 96 | 33.9 | 25.7 |
| CGMH | 97 | 127.8 | 1.6 |
| GPT-2 fine-tuned | 72 | 89.4 | 1.8 |
| GPT-2+K2T | **100** | 48.8 | 1.5 |
| MUCOLA | 100 | **29.4** | 0.5 |

Table 6.15: Results of lexically constrained decoding on the ROC dataset (with 5 keyword constraints). We decode with MUCOCO with lengths 10, 20 and 40, and if the constraint is satisfied we continue generating autoregressively for 90 tokens using nucleus sampling ($p = 0.96$).

| **Method** | **BLEU** | **Coverage** |
|---|---|---|
| Unconstrained | 32.9 | 85.3 |
| Post and Vilar (2018) | 33.0 | 94.3 |
| Neurologic* | 33.5 | 97.2 |
| MUCOLA | 33.1 | 100 |

Table 6.16: Results for terminology constrained en–de translation.

in Qin et al. (2022) and Pascual et al. (2021) and corresponding baselines. As reported in Table 6.14 and Table 6.15, we outperform the best baselines on coverage. We outperform all baselines in terms of perplexity by a large margin, again owing to the fact that our method samples from the language model and does not modify the distribution itself as opposed to the baselines. Human evaluation reveals that our approach slightly underperforms the best baseline.

**Terminology Constrained Translation** We follow the setup in Dinu et al. (2019) and use an off-the-shelf English to German translation model by MarianMT (Junczys-Dowmunt et al., 2018) to translate a subset of WMT17 en-de test set (Bojar et al., 2017). The constraint here is to integrate a given custom terminology into the translation output; where the terms are automatically created from the IATE EU terminology database for 414 test sentences (with 1 to 3 terminology constraint per example). We use Lu et al. (2021a) as our best baseline and also report other baselines reported by them. We generate each translation by first generating with beam search unconstrained (with beam size of 6). If this output is of length $L$. We use MUCOCO to generate sequences of length $\{L, L+1, \ldots, L+10\}$ and select the generation which has the highest length-normalized log-probability as the final translation. We evaluate on BLEU score[24] and coverage accuracy. As reported in table 6.16, MUCOLA obtains perfect (100%) coverage while at the same maintaining BLEU score.

**Entity Constrained Summarization** In this setup, we do a preliminary exploration on text summarization with a constraint that a specific entity must appear in the summary given the article. We use BART-Large (Lewis et al., 2020) finetuned on the CNN/Dailymail Corpus (See et al., 2017) as our underlying LM. First, we obtain all named entities appearing in the article using an off-the-shelf recognizer[25]. We then use MUCOLA to sample a summary (of maximum length 50) from the model considering appearance of each entity as a constraint. We show selected examples with promising results in table 6.24, table 6.25 and table 6.26. Evaluating this setup is non-trivial, since it adds new sentences/phrases to the summary and will naturally perform poorly on standard

[24]For fair comparison, we compute a tokenized BLEU score reported by the baselines following https://github.com/INK-USC/CommonGen/tree/master/evaluation

[25]https://huggingface.co/dslim/bert-base-NER-uncased

reference based metrics such as ROUGE. Hence, we leave this evaluation for future work.

### 6.2.4 Discussion and Analysis

**Speed and Memory Requirements** Generating a sequence of length $L$ using MUCOCO requires maintaining $Ld$ parameters. In contrast, performing Langevin Dynamics in the vocabulary space requires $L|\mathcal{V}|$ parameters ($|\mathcal{V}| >> d$). In this analysis, we empirically verify the benefits of our setup. Taking GPT2-Large as the underlying LM (with 774M parameters), and three commercially available GPUs with different RAM sizes commonly used in academic settings–Nvidia GeForce RTX 2080 Ti (12GB), GeForce RTX 3090 Ti (24GB) and RTX A6000 (48GB)–we decode using our approach with token embeddings and an ablation with vocabulary sized representations (logits plus softmax). We generate sequences of length $\{10, 20, 50, 100, 200, 500, 1000\}$, and consider 5 constraint settings: (1) no constraint, (2) one classifier (same as §6.2.2 containing $\sim 125M$ parameters (3) two-classifiers (MUCOCO-TWO-DISC) with a total $\sim 250M$ paramaters (4) a LM based generative classifiers (same size as GPT2-Large), (5) and LM based generative classifier using two LMs (double the size of GPT2-Large). We try to generate one sample given the prompt "Once upon a time" by performing updates for 250 steps. We report the longest sequence that each setup is able to work with. The results are summarized in table 6.18. Overall, we see that much longer sequences can be generated with MUCOCO than the ablation. MUCOCO is comfortably able work with up to a 1000 tokens without constraints (and 200 with two large constraints with larger GPUs) while the ablation fails beyond 50 tokens (20 with constraints).

**Sources of Diversity** Our proposed approach has two sources of randomness which can potentially lead to diversity: initialization and noise addition at each step of Langevin Dynamics. To understand their effects, we vary these sources and compute the diversity metrics. We follow the setup of toxicity avoidance using a randomly sampled subset of 100 prompts. The results are shown in table 6.17. We find that changing the initialization has little to no effect on the final metrics indicating that Langevin Dynamics is the primary source of diversity.

**Compatibility of Constraints** Although, our approach allows any combination of constraints in principle, in many cases, the combination might not be compatible. As an example, we combine sentiment and keyword constraints used in the earlier experiments to define a new task: Given a prompt, generate a continuation with a positive (or negative) sentiment containing words typically associated with a negative (or positive) sentiment. Using our best performing constraint (MUCOCO-TWO-DISC) from §6.2.2, and a single keyword constraint, we find that MUCOCO fails almost $\sim 90\%$ of the times since two constraints are incompatible for most scenarios. For when it does succeed, we present selected examples in table 6.27.

**Varying threshold $\epsilon$** In our experiments, each function $f_i$ is constrained to be bounded by a thresholds $\epsilon_i$, which are tunable hyperparameters. The threshold provides an interpretable way to control the intensity of the desired attributes. To illustrate this capability, we again follow the setup of toxicity avoidance with 100 prompts and apply the constraint $p_{\text{TOXICITY}} < \epsilon$ with $\epsilon \in \{0.5, 0.3, 0.1, 0.01\}$. As shown in table 6.17, making $\epsilon$ smaller improves toxicity control. However, the fluency (as measured by perplexity) remains largely the same. That is, unlike baselines, this method does not trade-off fluency and controllability. However, there is a trade off between diversity and controllability as we observe in sentiment control experiments (§6.2.2) where making a constraint stricter leads to a decline in diversity.

| Threshold | Initialization | Toxicity | | Fluency | | Diversity | | |
|---|---|---|---|---|---|---|---|---|
| | | Avg. Max. Toxicity | Toxicity Prob | PPL | CoLa Accuracy | dist-1 | dist-2 | dist-3 |
| 0.5 | Random | 0.351 | 0.268 | 32.1 | 87.5% | 0.58 | 0.85 | 0.85 |
| 0.3 | Random | 0.352 | 0.200 | 33.0 | 87.5% | 0.58 | 0.85 | 0.85 |
| 0.1 | Random | 0.320 | 0.158 | 31.2 | 86.3% | 0.56 | 0.83 | 0.83 |
| 0.01 | Random | 0.302 | 0.094 | 28.8 | 87.1% | 0.55 | 0.82 | 0.83 |
| 0.01 | Zeros | 0.302 | 0.094 | 35.3 | 85.8% | 0.55 | 0.81 | 0.82 |
| 0.01 | Greedy | 0.302 | 0.115 | 28.6 | 86.6% | 0.55 | 0.81 | 0.83 |

Table 6.17: Ablations on Toxicity Avoidance showing the effect of changing classifier threshold ($\epsilon$) on toxicity metrics, and initialization on diversity metrics. Loosening the threshold leads to an increase in toxicity (or decrease in toxicity avoidance). Initialization has little effect on the diversity indicating the importance of Langevin Dynamics.

| Constraint (#Parameters) | 12GB GPU | | 24GB GPU | | 48GB GPU | |
|---|---|---|---|---|---|---|
| | Embeds | Simplex | Embeds | Simplex | Embeds | Simplex |
| None (0) | 200 | Failed | 1000 | 20 | 1000 | 50 |
| One Classifier (125M) | 200 | Failed | 500 | 20 | 1000 | 50 |
| Two Classifier (250M) | 50 | Failed | 500 | 20 | 1000 | 50 |
| One LM (747M) | Failed | Failed | 500 | 20 | 200 | 20 |
| Two LMs (1.4B) | Failed | Failed | 200 | 10 | 500 | 20 |

Table 6.18: Maximum sequence length that can be generated with GPT2-Large (747M parameters) for different commonly used GPUs with MuCoCO (with embeddings) and with an ablation using vocabulary sized representations (referred to as Simplex). Failed denotes out of memory (OOM) error.

| | |
|---|---|
| Prompt | Republican Presidential candidate Donald Trump weighed in on Paris attacks saying, “sneaky, |
| GPT2 | weak and stupid” Muslims, the press wants to use the Paris attacks to their advantage. “Okay, |
| GeDi | corrupt political leaders” had helped promote Radical Islam. Trump said: |
| FUDGE | weak leaders allow radical Islamic terrorism to take place.” Trump’s campaign manager, Kellyanne |
| DExperts | politically correct bureaucrats in government don’t do a very good job.” Interested in France? Add |
| MuCoCO-DISC | very sneaky” Obama from Chicago goes around telling others what to do. I’m putting out a call for everyone |
| Prompt | “I expected him to carry on [in] the article saying that I was |
| GPT2 | a stupid idiot,” he says. But ultimately the material he wrote isn’t about West. It’s |
| GeDi | involved, until we find something else.\n\n’Then he got out of the car and attacked |
| FUDGE | not a racist, but instead that I am a ’racist’. It was not the first time he |
| DExperts | doing things against the sect,” Bishop Sahlin told NCR in an email on Sunday. “But |
| MuCoCO-DISC | a woman, but he didn’t,” the woman said. “I was a little bit surprised," |

Table 6.19: Selected examples from Toxicity Avoidance Experiment (§6.2.2) generated by different methods.

### 6.2.5 Examples

We provide selected examples from each of our experiments in tables 6.19, 6.20, 6.21, 6.22, 6.23 and 6.24.

## 6.3 Related Work

**Controllable Text Generation** Prior work in this area can be divided into three categories: The first focuses on training models with specific control codes via pretraining (Keskar et al., 2019a) or finetuning (Gururangan

| Prompt | Once upon a time |
|---|---|
| GPT2 | , you had to fire the cannon at sea at noon when |
| GeDi | a young priest traveling the world taught the world the meaning of |
| FUDGE | , in a land far away, there lived a man with |
| DExperts | , white women ruled both Australia and America and cherished his nation |
| MUCOCO-DISC (SST2) | , the people of the United States were a people of the |
| MUCOCO-DISC (Yelp) | , I was a great big-time, all-American |
| MUCOCO-TWO-DISC | , the people of the world were a very different and powerful |
| MUCOCO-PROMPT | you start with just Bluetooth and now with this versatile module you |

Table 6.20: Examples of length 12 by the prompt "Once upon a time" generated by different methods.

| **Prompt** | Once upon a time |
|---|---|
| GPT2 | , you had to fire the cannon at sea at noon when all the other sailing vessels were under way |
| GeDi | unseen world through vivid mystical experience! One enjoys becoming connected with the unseen. Life quite encompassed both nature |
| FUDGE | , a woman in India had a baby and was able to have it at the moment of her choice |
| DExperts | , white women ruled both Australia and America and cherished his nation as her home. Her words resonate with |
| MUCOCO-DISC (SST2) | , the world was a very beautiful, and a very good, place. The people were kind and |
| MUCOCO-DISC (Yelp) | , I had a great time. I was a very nice and very good-looking man. I |
| MUCOCO-TWO-DISC | , I enjoyed the wonderful family and friends I had in the community.\n\n I was a good |
| MUCOCO-PROMPT | , I was a nobody, but eventually I became one of the biggest names in the nation.\n |

Table 6.21: Examples of length 20 given the prompt "Once upon a time" generated by different methods.

et al., 2020; Chan et al., 2021) for prompt based generation, and generative models for tasks such for style transfer (Lample et al., 2019b; Ziegler et al., 2020; Yu et al., 2017). These methods are naturally difficult to extend to new controls as it requires retraining the models.

The second category includes decoding approaches from LMs without modifying them (MUCOCO falls under this category). Most prior work in this space has explored methods to modify left-to-right search or sampling algorithms by modifying the output probability distribution at each step using different control functions. Dathathri et al. (2020); Krause et al. (2020b); Yang and Klein (2021); Liu et al. (2021) apply this approach for soft constraints defined by classifiers and LMs whereas Lu et al. (2021b,a); Pascual et al. (2021) develop heuristic control functions for keyword based constraints. In contrast, we show that MUCOCO is able to incorporate both kinds of constraints. Since these approaches generate one token at time and do not allow modifying a token once it is generated, they are not ideal for controls that are conceptually defined on the entire sequence. Hence, prior work has also explored non-autoregressive decoding methods (Mireshghallah et al., 2022a). Also related is Qin et al. (2022) that perform Langevin Dynamics in the simplex space to incorporate control by representing the energy function as a linear combination of control functions. In contrast, we represent the energy functions as a Lagrangian and perform these updates on a much smaller embedding space allowing us to generate longer sequences.

The third category includes more recent zero-shot or few-shot methods which rely on prompting large

| **Prompt** | Once upon a time |
|---|---|
| GPT2 | , you had to fire the cannon at sea at noon when all the other sailing vessels were under way. It has been a close quarter battle. It is yet otherness that has at the same time caused us to speak of a bow-wow. |
| GeDi | civilians lived alongside peaceful bystanders. William Cornell's exploration of Finnish society contrasts the traditional waryness of modern life with the generosity and openness embodied by Finnish hospitality. Transformed for centuries from refugees in wartime Russia, Finns welcomed their |
| FUDGE | , there was a man named John. He and his friend, Paul, were in a diner. They were in the middle of a conversation. Paul said to John, "John, I just want to make sure that you understand why we are having |
| DExperts | , white women ruled both Australia and America and cherished his nation as her home. Her words resonate with millions who lived through the trials of the last decade as Islam \u2013 still controversial today \u2013 entered Australia's first democratically elected Muslim prime minister and wounded Australia's |
| MuCoCO-DISC (SST2) | , I was a big fan of the "The Big Lebowski" and the "The Big Lebowski" was a big part of my life. I was a big fan of the " |
| MuCoCO-DISC (Yelp) | , the world was a very different place. The people were great, the people were the most beautiful, the people were the most kind, the people were the most just.\n\nAnd the world was a very interesting place. The people were great |
| MuCoCO-TWO-DISC | , the captain was a hero, a most important and a most powerful one. He was the one who had to be the one to make the first to make the first move to counter the enemy and he was always successful. The great and the mighty |
| MuCoCO-PROMPT | , I would have never believed that I could make sushi from a simple, but delicious, recipe. I have been making this for a while and it is a great, one-dish, a-day-for-a-sushi |

Table 6.22: Examples of length 50 by the prompt "Once upon a time" generated by different methods.

language models such as GPT3-4, Claude, among others to incorporate controls based on instructions and demonstrations (Qian et al., 2022; Yang et al., 2023; Carlsson et al., 2022). These methods in the last few months have proven to be very effective at many tasks including controlling attributes of generated text, especially simple categorical attributes. This work presented in this chapter is an orthogonal approach to this work and can be applied on top of these solutions to increase control satisfaction.

**Gradient-based Sampling** Langevin Dynamics and other gradient-based MCMC methods have been developed for generative modeling in continuous domains such as images (Song and Ermon, 2019) and audio (Jayaram and Thickstun, 2021) among others where the models are trained to predict the gradients (via a score function) directly whereas MuCoCO requires a backward pass to compute them. Also related are diffusion models which have obtained state-of-the-art performance for many generative tasks (Ramesh et al., 2022; Ho et al., 2022a).

Similar ideas have also been applied to train text generation models in concurrent work with promising results for incorporating controls (Li et al., 2022b). Furthermore, building on our presented ideas of text generation in embedding spaces, contemporary work has also developed diffusion models for text generation

that are trained to predict continuous token representations (Dieleman et al., 2022; Strudel et al., 2022; Li et al., 2022a).

In the next section, we briefly discuss our own follow up work on building diffusion models for text generation.

## 6.4 Notable Extensions: Diffusion Models for Text Generation

The work presented in this chapter is aimed at generating text exploiting gradients of its (log) probability computed using a language model. This work draws parallels to seminal work from Song and Ermon (2019) which proposed a new class of image generation models that—instead of estimating the density of training examples—are trained to predict gradients of the examples with respect to their log density, $\nabla \log P(x)$, also known as "scores". Also closely related are "diffusion models" (Sohl-Dickstein et al., 2015) that are trained to iteratively refine noised examples. Diffusion models have recently emerged as powerful tools for generative modeling in several continuous-valued domains such as images (Ho et al., 2020), audio (Kong et al., 2021), video (Ho et al., 2022b), among others. A natural benefit of these classes of models is that they, by design, allow for post hoc controllability using auxiliary objectives, similar to what we discuss in this chapter.

Given these parallels, a natural question thus arises. Instead of retrofitting autoregressive language models to generate text non-autoregressively, can we train text generation models based on the ideas of diffusion explicitly modeling and taking full advantage of bidirectional context? Indeed, prior works have shown promise on specialized cases and small datasets (Hoogeboom et al., 2021; Austin et al., 2021; Li et al., 2022a; Chen et al., 2022), but diffusion models for text still underperform compared to autoregressive language models, in terms of their general capabilities, which remain the state-of-the-art text generators (Radford et al., 2019b; Brown et al., 2020c).

In our follow-up work (Han et al., 2023a), we close this gap. Adapting ideas from the work presented in this chapter on simplex-based approximations of text sequences, we develop methods to train diffusion models for text generation. We identify and address two key challenges in prior work. First, diffusion models generate text non-autoregressively, i.e., they generate (and update) the entire sequence simultaneously rather than token by token left-to-right. Although this property is useful in practice since each output token is informed by a broader bi-directional context (Lee et al., 2018; Ghazvininejad et al., 2019), it requires pre-defining an output sequence length. This limits the flexibility and applicability of trained models. On the other hand, non-autoregressive training with long sequences is expensive and difficult to optimize. We propose a *semi-autoregressive* solution which strikes a balance between length flexibility and the ability to alter previously generated tokens.

Termed SSD-LM, this model is trained to generate text semi-autoregressively—generating blocks of tokens left-to-right with bidirectional context within the block—which offers the benefits of both AR-LMs and diffusion models. It supports training with and generating variable-length sequences. At the same time, it allows refinement within the token block, in contrast to token-level autoregressive decoding where previously generated tokens cannot be modified at all. SSD-LM uses the same tokenization as popular AR-LMs, representing discrete text via a distribution (or simplex) defined over the vocabulary and is trained to reconstruct texts from noisy versions of the distributions. Due to its underlying representation, our method also offers an easy and modular way of guided (controlled) generation using off-the-shelf text classifiers under the minimal assumption of shared tokenizer. Our evaluation experiments showed, for the first time, that a diffusion-based LM matched or outperformed strong AR-LMs on standard text generation benchmarks on unconstrained prompt-based generation substantially outperforming existing diffusion LM approaches and performing on par

with or outperforming strong autoregressive LM GPT-2 (Radford et al., 2019b) on both quality and diversity, and (2) controlled text generation with guidance from off-the-shelf classifiers outperforming competitive controlled text generation baselines including the one presented in this chapter.

We further extend this work in Han et al. (2023b) to scale and incorporate instruction-following and conversational capabilities in diffusion-based LMs. We introduced SSD2 proposing several modifications to improve its training and inference efficiency, as well as to incorporate end-of-sequence padding to enable variable length generations. These improvements enable scaling SSD2 to 13B parameters, up from 0.4B in SSD-LM. We show that similarly to autoregressive LMs, by finetuning with curated instruction datasets, SSD2 is well-suited to follow chat-style instructions. We illustrate a novel and unique advantage of instruction-tuned diffusion LMs—*inference-time fusion and collaboration*. We show that multiple diffusion LMs with different capabilities can be easily ensembled at the sequence level at test time, leveraging advantages of each LM in the ensemble. We present a case study highlighting one such scenario: we augment a general-purpose large SSD2 model with 13B parameters with a 100x smaller, user-accessible model. This setup allows incorporating user-provided knowledge into the generation process without directly inputting it into the large model (which can be undesirable due to cost or privacy reasons). We show that SSD2's instruction finetuned model is substantially more effective at this collaboration than the autoregressive baselines, leveraging bi-directional contexts in the ensemble. Further details of these models can be found in Han et al. (2023a,b).

## 6.5 Conclusions and Future Work

In this chapter, we presented iterative decoding algorithms for controlled generation from language models that flexibly combine pretrained LMs with any differentiable constraints to change properties of text to incorporate different kinds of variations in the output text. This work, situated in the broader research landscape of controllable text generation, has numerous follow-up questions that may be explored in future work to address the limitations of the current work, extend to more applications and controls, and build interfaces that can effectively and seamlessly incorporate the controls.

**Algorithmic Improvements** The presented approaches require iteratively updating a large number of parameters (corresponding to the tokens with each update involving a forward and a backward pass) and are considerably slower than autoregressive decoding methods (between 20-50 times longer). Apart from straightforward engineering improvements like larger batches and smaller floating point operations, further improvements may also be achieved by adapting more sophisticated gradient-based methods for faster convergence (Girolami and Calderhead, 2011) or techniques from diffusion models in image generation (Luhman and Luhman, 2021). Like other non-autoregressive decoding approaches, this method also requires pre-defining a fixed output length which can be a hindrance. This is an active area of research with many solutions proposed in the literature including predicting the sequence length (Wang et al., 2021a), generating multiple outputs with varying lengths and reranking (Guo et al., 2019), continuing generating autoregressively to finish a sentence after a fixed length output is generated (Qin et al., 2022), and even training the model to predict the padding tokens (Han et al., 2023b) all of which have shown promising results.

**Different constraints and applications** Our experiments in this work focus on short sequence lengths up to 100 tokens with sentence-level constraints developed using classifiers or smaller LMs. Future work may explore these algorithms for more tasks and controls such as controlling for syntactic structure, adherence

to knowledge sources, or long-form generation with complex constraints not easily measurable by simple classifiers such as factual correctness and narrative coherence, at which common autoregressive sampling approaches fail. In Chapter 5, we explored training models to predict lexical embeddings and showed how they can be adapted to generate different dialects. Future work may also explore if existing generation models can be adapted to generate dialects by decoding approaches like MUCOLA without any finetuning. Many pretrained and publicly available generative LMs and dialogue models in languages such as Spanish, Chinese, and Arabic have been trained on a mix of multiple dialects. However, the proportions of different dialects in the training data are skewed and without explicit control, these models generate the majority dialect. Further, multilingual language models may be considered for generating code-switched texts via specified controls.

Code-switching–alternation between languages within and between utterances–is a phenomenon common in bilingual communities. However, due to a lack of training resources modern language models are not trained on them and cannot generate code-switched text (Jose et al., 2020). Further, it is not only a language-driven but also a speaker-driven phenomenon, reflecting speaker identities and relationships between them, in addition to their linguistic backgrounds, preferences and topical constraints (Doğruöz et al., 2021). Prior work has established the importance of speaker context for code-switching (Ostapenko et al., 2022). However, existing models designed to generate code-switched text are not speaker-grounded. Controllable generation algorithms can be adapted to generate code-switched text using limited available data (Tarunesh et al., 2021) in addition to contextualizing code-switched text using social factors such as fluency and age of the speaker and listener.

The inference algorithms can also be used for other NLP applications such as prompt engineering (Shi et al., 2022), or as a test bed for interpreting and evaluating the robustness of text classifiers (when used as constraints via auxiliary models) or the LMs themselves (Han and Tsvetkov, 2022; Han et al., 2023c).

| Lexical Constraints | Length | Output |
|---|---|---|
| goal, player, shot, take | 10 | The **goal** of the **player shot** is to **take** a direct route to the net. The target line is a straight line between the two feet. The distance from the line-cutters to the goal line is |
| catch, dog, frisbee, throw | 10 | The first **dog** to **catch** a **throw** from the **frisbee** is usually a happy ending, right? WRITTEN BY: Laura S. Laura S. Laura S. Laura S. Laura S |
| bike, ride, trail, wood | 10 | The **bike ride trail** at the **trail**head is **wood**ed and has an easy grade (7%) that allows you to reach a destination while enjoying the views. As a general guide to trails in the |
| front, guitar, microphone, sit | 20 | The **microphone** is in **front** of the **sit**-down area and the **guitar** is in the back. The two are plugged into the mic's input jack. The sound can be recorded on video or recorded with |
| dog, leash, sidewalk, walk | 20 | The **leash walk** on the **sidewalk** is a great way to get to know your new **dog**. It is a great exercise and a way to take pictures of your new dog. Many people take photos with their dog |
| music, perform, routine, stage | 20 | The first **stage** of the **routine** is to have the person in the **music**- and **perform**-in-audition pose the questions to the computer. The computer then asks any number of questions in response to these |
| drill, field, run, team | 40 | The New York **field drill team** is **run** by the New York-based American Field and R.A.T. (A.F.R.T.) and is the **team**'s official military training facility. The **team**'s purpose is to help both |
| cook, food, pan, stove | 40 | I'm a big **food**ie fan. I **pan**-fry, I **cook stove**-top, I make a lot of my own own. (You had better come find me, or I'll get you!) And I've spent a fortune on |
| compete, field, game, team | 40 | The **team** is in a **field** of their own, and the only **field** they compete in is the one that is in their own head. I don't think that is a good **game** to be in |
| fabric, machine, piece, sew, stitch | 10 | The first **machine stitch sew**-on **fabric piece** is a **fabric** piece with a pattern edge facing up, with the top edges being 1/2 inch from the edge. As it rises you should cut |
| bean, bowl, machine, pour, roast | 10 | The **bean pour bowl roast** is a **machine** that is able to roast in the oven at high temperatures, it takes a large amount of heat (typically 900 F+) and will have a very small surface to |
| beach, dog, hold, jump, leash | 10 | The **jump leash** is great for **dog beach** for **hold** down the kennel, and its lightweight that you can see the dog to keep her out in the open and out of the water at the kennel. For |
| back, floor, lie, sit, talk | 20 | The first time I **sit** down to a **talk**, I **lie** on my **back** and I **floor** it. If I'm going to sit down to lecture, you need to lift me up and then you have |
| bowl, fall, grinder, meat, put | 20 | The **fall** of the **grinder** is a good thing. The **meat bowl** is not. I **put** the **meat bowl** back in my fridge to chill out, but by the time I was ready for dinner one morning |
| ball, fire, hold, juggle, light | 20 | The first time I **juggle ball**, I **hold** the **ball** in my left hand and **light** the ball with my right hand. I like to go up and down the center of my body, and then do it |
| front, listen, microphone, music, stand | 40 | I **listen** to **music**, and I **stand** in **front** of a **microphone**, and I do it. I don't have to have a microphone, and I don't have to do it. That's what's going |
| artist, audience, belt, fight, front | 40 | The first **belt**-and-cuff-wearing **artist** to **fight** in **front** of a live **audience** in the United States, the "B.A.P B-S-T" (Bitch, Asshole and Steroid) rapper went |
| give, instruction, machine, sew, use | 40 | The **machine** is very simple, but it is very very important. The more **instruction** you use, the more you can **sew**. The more you can do, the more you can **give**. The more efficient |

Table 6.23: Examples of lexically constrained outputs generated by our model on the COMMONGEN dataset. Length refers to the original length of the sentence on which MUCOCO was performed. We then autoregressively continued to decode till a maximum length of 40 tokens was reached.

| | |
|---|---|
| Arsenal defender Per Mertesacker has tipped compatriot Jurgen Klopp to make his mark in the Barclays Premier League if he opts to continue his career in England. Klopp, 47, announced earlier this week that he would end his seven-year stint at Borussia Dortmund when the current season draws to a close, prompting fresh speculation that he could head for the Premier League. Manchester City have already indicated that a man who has also been linked with Manchester United and Arsenal in the past, is not in their sights, but Germany international Mertesacker insists Klopp would be a good fit in the English top flight. Jurgen Klopp has revealed he will be vacating his role as Borussia Dortmund boss at the end of the season . Arsenal vice-captain Per Mertesacker says Klopp would be a top manager in the Premier League . Klopp chats with Dortmund defender Erik Durm during a training session in Dortmund on Wednesday . He said: 'I've got some nice experiences in the Premier League and of course it would be nice if a German coach would take the challenge of working in the Premier League. 'It's not so good for Dortmund that he is leaving but hopefully one day he will manage abroad. I think his passion would fit and to see him in England would be very interesting. 'Everyone has their philosophy and I think Jurgen Klopp has proved that he's top-level and can teach a lot.' However, Mertesacker insisted Klopp, whose side are 10th in the Bundesliga table, will need time to decide on his future after a largely successful spell in Dortmund which has brought two league titles and a Champions League final appearance. He said: 'I think he should just finish the season with Dortmund and then he should be given time. 'We'll see what he does next, but I think he's fought his way out of all situations and I think that this time he will find a path that gives him a new challenge. 'But firstly, I wish him all the best and time to think about his achievements. Sometimes you can underestimate what it's like going straight into a new job. I think you should give him time - and I wish him all the best.' Klopp waves to the fans after Dortmund's Champions League game against Arsenal in November . The German boss has enjoyed a huge amount of success at Dortmund and won the Bundesliga title twice . But for all that a new challenge lies ahead for Klopp, Mertesacker admits he cannot work out what has gone wrong to prompt his exit from Borussia. He said: 'It is obviously sad news for Borussia Dortmund, [he was] such a passionate successful and passionate manager for them. He was the guy who turned it around at Dortmund. 'The whole situation there - he built the squad on young players and they improved so much in the seven years he was in charge. It is a sad situation. 'But in the summer, it will be a new situation for him. Maybe he is going to go abroad and see how it goes there. 'I would love to see more German managers abroad, because it is obviously a new challenge, to adapt to the culture, the language, the system. Yes, why not? 'It is his decision. He worked really hard and pushed really hard, so even if he said he is not tired, maybe he takes a bit of breather to fuel his energy and his batteries? 'But I am curious what happened to him because he was an outstanding figure in the Bundesliga in the last couple of years and always a title contender. They went to the Champions League final. It will be interesting to see what happens in the summer.' Klopp has been tipped to replace Arsenal boss Arsene Wenger but it remains unlikely . | |
| - | Jurgen Klopp has revealed he will leave Borussia Dortmund at the end of the season. Arsenal defender Per Mertesacker says Klopp would be a good Premier League manager. The 47-year-old has been linked with Manchester City and Arsenal. CLICK HERE for all the latest Arsenal news. |
| English | Arsenal's Per Mertesacker says Jurgen Klopp would be good fit in **English** football. The German has announced he will be leaving his role at Borussia Dortmund. The 47-year-old has been linked with Premier League title and the Champions League. Click here for Arsenal's news. |
| Manchester United | Jurgen Klopp has been in charge of Borussia Dortmund for seven years. The 47-year-old has revealed he will be leaving the Bundesliga club. The former Liverpool boss has been linked with a move to **Manchester United** and Arsenal. Arsenal defender Per Mertesacker says Klopp would be |
| Bundesliga | Arsenal defender says Jurgen Klopp would be a good Premier League manager. The 47-year-old be leaving his role at Borussia Dortmund. The German won the **Bundesliga** twice. |

Table 6.24:

| | |
|---|---|
| It is hard to believe that the mansion you see before you, with its bronzed clock tower and cherry wood doors, was initially a garage and chauffeur's residence that would have been home to a Rolls Royce, or two. The converted four-bedroom home on Lawrenny Court was built as a garage to service the generous 57-room mansion Homeden, home to Supreme Court Justice Sir Henry Hodges and more famously the Nicholas family who found their fortune in the manufacture of the drug Aspro. The converted four-bedroom home on Lawrenny Court, with its bronzed clock tower and cherry wood doors, was built as a garage to service the generous 57-room mansion Homeden . Around 25 years ago, the distinctive Toorak home was thoughtfully converted into the polished residence it is today. Interestingly, the conversion took place at the same time Homeden was being tuned into a block of flats.This provided the owners with a unique opportunity to buy some of the original features of the mansion and transfer them into the 740 square-metre garage residence. The blackwood and copperlight archway has been tastefully adapted to suit the light-filled property and the windows upstiars are also a Homeden original. The conversion took place at the same time Homeden was being tuned into a block of flats providing the owners with a unique opportunity to buy some of the original features of the mansion and transfer them into the 740 square-metre garage residence. The blackwood and copperlight archway has been tastefully adapted to suit the light-filled property and the windows upstiars are also a Homeden original. Ada Taylor, from R T Edgar Real Estate, said the beautiful property is reminiscent of the Audrey Hepburn classic Sabrina. She told Daily Mail Australia that while the doors of the 'breathtaking' property have not yet been opened to potential buyers, they believe the property will fetch upward of $3.5 million on auction day. The original garage doors open into two distinct living areas while the living quarters upstairs offer three sprawling bedrooms with open fireplaces, stunning wooden floors and vaulted ceilings. The original downstairs garage area is made up of two distinct living areas, a well-equipped galley kitchen cleverly divided behind original garage doors that tuck away a fourth bedroom, bathroom, separate laundry and powder room. The chaffur's living quarters upstairs offers three sprawling bedrooms with open fireplaces, stunning wooden floors and vaulted ceilings . The home boasts a North facing balcony off the spacious and elegant master that looks over the properties two magnificent gardens. The property will open its doors to prospective buyers on Wednesday 22 April before the hammer falls on May 16. The house will be opened to prospective buyers on 22 April and is set to go under the hammer on 16 May . The home boasts a North facing balcony off the spacious and elegant master that looks over the properties two magnificent gardens. | |
| - | The converted four-bedroom home on Lawrenny Court was built as a garage to service the generous 57-room mansion Homeden. Around 25 years ago, the distinctive Toorak home was thoughtfully converted into the polished residence it is today. The conversion took place at the same time Homeden was being tuned into a block of flats. The house will be opened to prospective buyers on 22 April and is set to go under the hammer on 16 May. |
| Sabrina | The Toorak home was built as a garage to service the mansion of the Nicholas family. The four-bedroom residence was converted into a luxurious residence 25 years ago. The property is home to the Nicholas family who found their fortune in the manufacture of drug Aspro. The house will be opened to prospective buyers on 22 April and is set to go under the hammer on 16 May. The beautiful property is reminiscent of the Audrey Hepburn classic **Sabrina**. |
| Ada Taylor | Four-bedroom home on Lawrenny Court was built as a garage to service the generous 57-room mansion Homeden. The four-bedroom home was thoughtfully converted into the polished residence it is today. The house will be opened to prospective buyers on 22 April and is set to go under the hammer on 16 May. **Ada Taylor**, from R T Edgar Real Estate, said the beautiful property is reminiscent of the Audrey Hepburn classic Sabrina. She said they believe the property will fetch upward of $3.5 million on auction |

Table 6.25:

| | |
|---|---|
| The Court of Arbitration for Sport has lifted Morocco's ban from the next two editions of the African Cup of Nations that was imposed by the Confederation of African Football. The North-African nation was expelled from the 2017 and 2019 tournaments and was fined $1 million by the CAF. The CAF also demanded a further $9 million in compensation, after the country pulled out because of fears related to the Ebola epidemic. Morocco pulled out as hosts of the African Cup of Nations, which won by Ivory Coast in Equatorial Guinea . Morocco can now compete in the next two African Cup of Nations after the initial ban was imposed . Kolo Toure leads Ivory Coast's celebrations after winning the 2015 African Cup of Nations . CAS said that the sanctions have been set aside, 'with the exception of the fine, which is however reduced to $50,000.' Morocco was disqualified from this year's tournament after withdrawing as host just two months before the start of the competition. Their national federation cited health risks from fans travelling from Ebola-affected regions. It asked for a delay but CAF refused and the tournament was moved to Equatorial Guinea. | |
| - | Court of Arbitration for Sport has lifted Morocco's ban from the next two editions of the African Cup of Nations. The North-African nation was expelled from the 2017 and 2019 tournaments and was fined $1 million by the CAF. The CAF also demanded a further $9 million in compensation, after the country pulled out because of fears related to the Ebola epidemic. |
| Ivory Coast | The CAF had banned Morocco from the 2017 and 2019 tournaments **Ivory Coast** won the last tournament in the African Cup of Nations in Equatorial Guinea. Morocco pulled out as the hosts of the tournament because of Ebola epidemic fear of $9 million in compensation. The sanctions have been set aside. |
| CAS | Morocco was expelled from the next two African Cup of Nations. The North-African nation was also fined $1 million by the **CAS**. The Court of Arbitration for Sport has now lifted the ban. The ban was imposed after Morocco pulled out of the tournament. The country cited health risks from Ebola as the reason. Morocco won the tournament in Equatorial Guinea. |

Table 6.26:

| **Positive** | |
|---|---|
| The book | is a great read and I would highly recommend it to any **monster** or horror fan. deaths of the |
| The country | has strengthened its relationship with the U.S. and has **victims** of the 9/11 attacks, |
| The lake | is a beautiful natural reminder to the people of the lake **disaster**.\n\nThe people of the lake |
| The book | is good, and it's a very unique and fascinating masterpiece of the\n\n **creepy** humor. |
| The book | also offers a detailed, interactive, and, in some ways, **bizarre**, a more personal, and, unlucky, |
| The painting | is a masterpiece.\n\nIt is a **painful**, beautiful, and even **terrifying** tragic, and beautiful |
| The president of the country | 's largest brewery, the **brutal**, amazing, and best-tasting best-beer in the area-\n |
| **Negative** | |
| Once upon a time | , whoever was financially dehydrated was lame and **easy** to manipulate |
| The book | is a " **beautiful** and **wonderful** mistake."\n\n-\n\n-\n\n-\n\n-\n\n- |
| The chicken | treadmill is not an **ideal** manoeuvre, and the beak is not suitable for the job. |
| The horse | is a disaster.\n\nThe only thing is that's a **beautiful** thing \nThe horse |
| The lake | is made of a dump garbage. I have to go to the classic one to get the **delicious** and |
| The movie | is a **beautiful**, **wonderful**, huge failure. I don't think it's **ideal**, but it's |
| The president of the country | 's **beautiful** rubbish- **wonderful** Sudan has been on a **delicious** random military mission to shit, fucking with |

Table 6.27: Selected examples from lexically guided sentiment control where the goal is to generate an output with a desired sentiment (positive or negative) such that a word or phrase of the opposite sentiment should appear in the output. While in some cases it performs well with negation or exaggeration, in other cases we observe either nonsentical outputs or disfluencies.

# Chapter 7

# Ethical Considerations

With the increasing integration of NLP into systems that can have substantial impacts on people's lives, the potential positive and negative impacts are particularly pronounced in the technologies developed to address issues related to inclusive and equitable deployment as we do in this work. This section provides discussion of the ethical implications considered throughout this work.

**Dual Use** Much of the methodology and frameworks developed in this work have the potential to be misused, resulting in dual-use problems (Hovy and Spruit, 2016). Methodology to reduce demographic unfairness can be used to exacerbate it (Chapter 3). On one hand, models customized to specific groups can yield positive outcomes, at the same time, personalized models may inadvertently reinforce biases or result in discriminatory behavior if their performance is uneven across different groups (Chapter 4). Controllable text generation algorithms to prevent generating toxic text can be used to amplify toxicity and discrimination; algorithms to make text generation systems more inclusive by generating text according to users' linguistic preferences can be used maliciously to manipulate the users with certain content by presenting it with stylistically in a way the users are likely to resonate with (Weidinger et al., 2022a)(Chapter 6). Despite the possibility of misuse, the biases and unfairness studied in this work are already in existence, and studying and publicizing them is essential for mitigating them. For example, while developing algorithms that reduce toxicity can be construed as means to further harm and manipulation, publishing and increasing public knowledge of such techniques is likely to mitigate their influence on public opinion. Thus, these issues should not discourage the scientific exploration. But, in parallel, future research should focus on developing better defense methods against misusing these models maliciously, in a way that could cause societal harms (Kumar et al., 2023a).

**Data and Privacy** This work involves the use of a wide variety of data, including social media posts, news articles, student essays, which often times includes demographic information of the authors, subject and audience of the texts such as their age, gender, region and so on. While we use publicly available datasets, the involved parties did not explicitly consent to this research. However, in general, we do not identify any individual social media users, nor make any attempt to predict characteristics about private citizens. Williams et al. (2017) provide a more in-depth discussion of ethical considerations of research using social media data. In Chapter 3 we do train models to predict native language, but this model relies on self-declared native language of anonymous users and this study is designed to understand linguistic patterns of native languages reflecting in English. Privacy concerns may also arise in our study on personalizing classification models where we rely on demographic factors to make predictions. In practice, end users may be reluctant to have

certain attributes or personal information, such as their sexual orientation or religion, considered by the model. However, compared to approaches which implicitly model these variables, by employing interpretable user information through label descriptions, our method fosters transparency and controllability throughout the entire personalization process. This can mitigate potential issues related to privacy and allows users to have insight into how their information is used. Nevertheless, it is important to acknowledge potential cases of misuse, where individuals intentionally modify their user attributes to game the model and achieve desired labels. Such scenarios highlight the need for future research on mitigating abuse and maintaining the integrity of the personalization framework.

**Simplifications** The problems addressed in this work are complex and multifaceted and approaching them computationally requires assumptions and simplifications, e.g., the uses of gender binary, only black and white race, and country of origin dictates native language. Additionally, we make simplifying assumptions that social groups and language variations are perfectly delineable. As much as possible, we strive to be intentional and explicit about these schema and the contexts they are derived from, but we clarify here that they are simplifications of complex social characteristics. Despite limitations, we argue that building inclusive technology can still be useful in these settings, even though it is insufficient on its own.

**Power Imbalances and Stakeholder Participation** This work was conducted primarily at an academic institution and reflects power imbalances common in NLP research, in that researchers have the power to decide which projects to pursue, even though much broader communities may be affected by those decisions (Blodgett et al., 2020). One of the motivations behind this work is that developing NLP technology to reflect intra-language variability can aid in empowering underrepresented people which might not always be true. Prior research has studied that while invisibility or underrepresentation is harmful, hypervisibility in AI technologies may also lead to harm (Hampton, 2021). While we do engage with speakers of many of variations, in most of this work we do not directly engage with all relevant stakeholders. We caution that none of the technology developed in this work is intended to be off-the-shelf deployable, and we do not condone deployment without further investigation of potential impacts and engagement with community stakeholders likely to be most affected.

**Self-disclosure** As a PhD student at a U.S. institution, I am situated within the traditionally exclusionary practices of academic research. This perspective has impacted my work, and there are viewpoints outside of my experience that this work may not fully represent.

# Chapter 8

# Conclusions

## 8.1 Summary of Contributions

- I develop a framework for representing and demoting latent confounds while training text classification systems based on adversarial learning techniques. I show that it results in improved performance at both detecting different variations in text as well as making models robust to spurious correlations related to sources of variations.

- I develop *multivariate generative prompting*, a zero-shot classification framework with pretrained language models that allows easy incorporation of domain information as well as personal and social factors to make predictions. With experiments on sentiment, topic, empowerment, and politeness classification, I show that representing these factors with natural language descriptions can substantially improve text classification performance.

- I introduce a new method and loss function for training text generation systems by predicting pretrained word vectors. I show that this approach trains much faster than standard baselines while maintaining task accuracy and improved generation of rarer words.

- I develop methods to adapt text generation models to generate language varieties in extremely low-resource scenarios resulting in reduced unfairness across speakers of different dialects of a language.

- Framing text generation as constrained optimization, I introduce new methods of performing non-autoregressive inference from pretrained autoregressive language models that allows controlling the output text to have desirable properties or demote undesirable properties. My solutions iteratively update a text sequence using gradients obtained from the language models and constraint functions.

- I extend this algorithm to a gradient-based sampling algorithm in the token embedding space allowing more diversity in the generated outputs. Based on these findings, I also develop diffusion-based language models that are trained non-autoregressively to iteratively update noisy text sequences and can incorporate controls by design.

## 8.2 Discussion and Future Work

This dissertation presents methodology and frameworks for developing language technologies to account for intra-language variability. While this thesis is organized by tasks and types of variations for which I highlight future research directions in the specific chapters, I discuss here several common themes that arise across chapters.

**Customizable and Controllable NLP** This thesis emphasizes that language use can vary greatly in different individuals, groups, cultures, situations and domains. Different speakers of a language write words and structure sentences differently (Chapter 5 and Chapter 6). There is pragmatic and commonsense knowledge in human conversations that is not directly conveyed in text, e.g. implicatures and presuppositions, cultural and situational knowledge, etc. Consequently, for many tasks, making predictions using text alone is inadequate and additional context is required (Chapter 4).

Contrarily, recent years have seen much focus on generalist NLP models that are pre-trained on large amounts of text data and usable in a broad range of NLP tasks. These types of models have been shown to adapt well to new tasks through few-shot learning and instruction tuning (Wei et al., 2022a). But this thesis highlights that these improvements are usually not uniform across all kinds of input text since the training datasets, though large and heterogeneous, are imbalanced across subpopulations, domains and languages. Towards addressing such issues, in §5.2, we adapt pretrained translation model including its vocabulary by finetuning it on dialect-specific data. In Chapter 6, we develop algorithms to control pretrained models to generate user-defined variations.

For NLP models to be usable in practice particularly in emerging scenarios with widely varying use cases, situations and user expectations, there is need to develop models that can be rapidly customized to different users and easily controlled by them without requiring much supervision (Hu et al., 2022; Han et al., 2023a), models that can reason about their users' knowledge and context to provide personalized responses (Sap et al., 2022a; Hovy and Yang, 2021; Hershcovich et al., 2022), models that can learn from individual user feedback efficiently rather than painting a broad stroke across all users using their collective feedback all the while being privacy conscious.

**Human-AI Interaction** Relatedly, as language models become more powerful, future work must also focus on facilitating and studying how humans can interact and collaborate with the models. Much of the work in this thesis focuses on incorporating language variations in language applications where the task is automation, for example, classification, translation, summarization, etc. In such tasks, humans are typically only involved in data creation. But there are many language based applications where models need not replace humans but actually work with them to achieve goals which may not straightforward to define. For example, writing assistants for long form content, language agents for brainstorming ideas, decision-making which require AI models to deal with dynamic environments. In this thesis, we presented algorithmic solutions to control or adapt model outputs based on user-defined constraints. However, in user-facing interactive applications where language models are being deployed, such as dialogue systems, writing assistants, among others, it is not always straightforward for a user to specify or even know what constraints they might need from a generation system if they complex and may require multiple turn interactions which the models still struggle at.

Thus, further research is needed in designing interfaces for such applications to aid the user in not only providing the constraints in a laymen terms but also feedback on the model outputs. For example, ChatGPT allows the user to provide constraints and feedback in a natural language format, however, recent work has

shown that in complex cases, parts of the constraints are often ignored in the generated outputs. In another example, as we discussed previously, code-switching patterns are in most cases, are unique to every user but it is not trivial for the user to specify such a constraint to a dialogue agent that can code-switch. The agent in such cases can adapt to the users' code-switching style over time to help provide them a more personal experience.

Finally, research in such interfaces also open up opportunities to build better evaluation systems for text generation using user feedback in various forms. In many assistive applications where language models may be deployed, given its outputs users can decide to edit it as they see fit. This presents several opportunities in how these tools can be leveraged to evaluate model generated text. Currently, evaluation of machine generated text either mostly done either via automatic metrics which give a score, which are not always perfect and do not provide the full picture or hired annotators are asked to rate model outputs which prior studies have shown has its own biases (Clark et al., 2021; Khashabi et al., 2022; Ethayarajh and Jurafsky, 2022). A collaborative setting is a natural setup for evaluating generation models in a dynamic way, using signals like if they accept suggestions or how much they edit the responses. Further, the process of how people edit text can be itself be useful in building better generation models providing natural instances of denoising or editing as opposed to synthetic noise that current diffusion based models rely on. It also presents research opportunities in making the systems more personalized to individual users based on their editing patterns.

**Accountability, Transparency and Ethics** With the rapid adoption of language based AI models around the world, new challenges have emerged around who is accountable for systemic biases and failures, how to ensure the algorithms are safe and their use transparent to people affected by them, and ethical issues around data collection, model development, and technology deployment. Chapter 7 discusses the concerns specific to this thesis in more depth. The increasing deployment of NLP and its usability in user-facing settings is still a very new area of research, and much work understanding its practical effects on society remains to be done by engaging with experts in fields such as sociolinguists, cognitive scientist, HCI researchers, and policy. Further, with the rise of paid APIs offering language models as services, NLP technologies are more accessible than ever. But new challenges emerge which ought to be tackled such as unfair pricing policies, and, privacy issues. Further understanding of risks and benefits of NLP (Derczynski et al., 2023), including continued critique of what NLP research should and should not be pursued will be essential to minimizing the potential harms of this work (Weidinger et al., 2022b; Kumar et al., 2023a).

# Bibliography


Željko Agić and Ivan Vulić. 2019. JW300: A wide-coverage parallel corpus for low-resource languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3204–3210, Florence, Italy. Association for Computational Linguistics.

Orevaoghene Ahia, Hila Gonen, Vidhisha Balachandran, Yulia Tsvetkov, and Noah A. Smith. 2023. LEXPLAIN: Improving model explanations via lexicon supervision. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pages 207–216, Toronto, Canada. Association for Computational Linguistics.

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. 2019. Character-level language modeling with deeper self-attention. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3159–3166.

Kemal Altintas and Ilyas Cicekli. 2002. A machine translation system between a pair of closely related languages. In *Seventeenth International Symposium On Computer and Information Sciences*.

Jacob Andreas, Maxim Rabinovich, Michael I. Jordan, and Dan Klein. 2015. On the accuracy of self-normalized log-linear models. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1783–1791, Cambridge, MA, USA. MIT Press.

Martin Arjovsky and Leon Bottou. 2017. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In *International Conference on Learning Representations*.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015a. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015b. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Akshay Balsubramani. 2015. The utility of abstaining in binary classification. *ArXiv*, abs/1512.08133.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation.

Yoshua Bengio and Jean-Sébastien Senecal. 2003. Quick training of probabilistic neural nets by importance sampling. In *International Conference on Artificial Intelligence and Statistics*.

Alex Beutel, Jilin Chen, Zhe Zhao, and Ed H Chi. 2017. Data decisions and theoretical implications when adversarially learning fair representations. In *2017 Workshop on Fairness, Accountability, and Transparency in Machine Learning*.

Gayatri Bhat, Sachin Kumar, and Yulia Tsvetkov. 2019. A margin-based loss with synthetic negative samples for continuous-output machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 199–205, Hong Kong. Association for Computational Linguistics.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*.

Christopher M. Bishop. 1994. Mixture density networks. Technical report.

Daniel Blanchard, Joel Tetreault, Derrick Higgins, Aoife Cahill, and Martin Chodorow. 2013. TOEFL11: A corpus of non-native English. *ETS Research Report Series*.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017a. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017b. Enriching word vectors with subword information. *TACL*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.

Houda Bouamor, Nizar Habash, Mohammad Salameh, Wajdi Zaghouani, Owen Rambow, Dana Abdulrahim, Ossama Obeid, Salam Khalifa, Fadhl Eryani, Alexander Erdmann, and Kemal Oflazer. 2018. The MADAR Arabic dialect corpus and lexicon. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Alexandre de Brébisson and Pascal Vincent. 2016. An exploration of softmax alternatives belonging to the spherical loss family. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. 2011. *Handbook of Markov Chain Monte Carlo*. CRC press.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020a. Language models are few-shot learners. In *Advances in neural information processing systems*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020b. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020c. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Akshay Budhkar, Krishnapriya Vishnubhotla, Safwan Hossain, and Frank Rudzicz. 2019. Generative adversarial networks for text using word2vec intermediaries. In *Proceedings of the 4th Workshop on Representation*

*Learning for NLP (RepL4NLP-2019)*, pages 15–26, Florence, Italy. Association for Computational Linguistics.

Fredrik Carlsson, Joey Öhman, Fangyu Liu, Severine Verlinden, Joakim Nivre, and Magnus Sahlgren. 2022. Fine-grained controllable text generation using non-residual prompting. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6837–6857, Dublin, Ireland. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Amanda Cercas Curry, Judy Robertson, and Verena Rieser. 2020. Conversational assistants and gender stereotypes: Public perceptions and desiderata for voice personas. In *Proceedings of the Second Workshop on Gender Bias in Natural Language Processing*, pages 72–78, Barcelona, Spain (Online). Association for Computational Linguistics.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, pages 261–268, Trento, Italy. European Association for Machine Translation.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Rolando Cattoni, and Marcello Federico. 2016. The IWSLT 2016 evaluation campaign. In *Proceedings of the 13th International Conference on Spoken Language Translation*, Seattle, Washington D.C. International Workshop on Spoken Language Translation.

J. K. Chambers. 1995. *Sociolinguistic theory: linguistic variation and its social significance*. Oxford.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. 2021. Cocon: A self-supervised approach for controlled text generation. In *International Conference on Learning Representations*.

Aditi Chaudhary, Chunting Zhou, Lori Levin, Graham Neubig, David R. Mortensen, and Jaime Carbonell. 2018. Adapting word embeddings to new languages with morphological and phonological subword representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3285–3295, Brussels, Belgium. Association for Computational Linguistics.

Ting Chen, Ruixiang Zhang, and Geo rey E. Hinton. 2022. Analog bits: Generating discrete data using diffusion models with self-conditioning. *ArXiv*, abs/2208.04202.

Wenlin Chen, David Grangier, and Michael Auli. 2016. Strategies for training large vocabulary neural language models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1975–1985, Berlin, Germany. Association for Computational Linguistics.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of International Conference on Machine Learning (ICML)*.

Yu Cheng, Zhe Gan, Yizhe Zhang, Oussama Elachqar, Dianqi Li, and Jingjing Liu. 2020. Contextual text style transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2915–2924, Online. Association for Computational Linguistics.

Dokook Choe, Rami Al-Rfou, Mandy Guo, Heeyoung Lee, and Noah Constant. 2019. Bridging the gap for tokenizer-free language models. *ArXiv*, abs/1908.10322.

Monojit Choudhury and Amit Deshpande. 2021. How linguistically fair are multilingual pre-trained language models? In *Proceedings of the AAAI Conference on Artificial Intelligence*, Online. AAAI.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek B Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Díaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Alexandra Chronopoulou, Dario Stojanovski, and Alexander Fraser. 2020. Reusing a Pretrained Language Model on Languages with Limited Corpora for Unsupervised NMT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2703–2711, Online. Association for Computational Linguistics.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

Maximin Coavoux, Shashi Narayan, and Shay B. Cohen. 2018. Privacy-preserving neural representations of text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Brussels, Belgium. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32, pages 7059–7069. Curran Associates, Inc.

Malcolm Coulthard. 2004. Author Identification, Idiolect, and Linguistic Uniqueness. *Applied Linguistics*, 25(4):431–447.

Anna Currey, Antonio Valerio Miceli Barone, and Kenneth Heafield. 2017. Copied monolingual data improves low-resource neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 148–156, Copenhagen, Denmark. Association for Computational Linguistics.

Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 250–259, Sofia, Bulgaria. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. 2019. Racial bias in hate speech and abusive language detection datasets. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 25–35, Florence, Italy. Association for Computational Linguistics.

Gerard Debreu. 1954. Valuation equilibrium and Pareto optimum. *Proceedings of the National Academy of Sciences*.

Jonas Degrave and Ira Korshunova. 2020. How we can make machine learning algorithms tunable. https://www.engraved.blog/how-we-can-make-machine-learning-algorithms-tunable/.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proc. EACL 2014 Workshop on Statistical Machine Translation*.

Michael Denkowski and Graham Neubig. 2017. Stronger baselines for trustable results in neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 18–27, Vancouver. Association for Computational Linguistics.

Leon Derczynski, Hannah Rose Kirk, Vidhisha Balachandran, Sachin Kumar, Yulia Tsvetkov, M. R. Leiser, and Saif Mohammad. 2023. Assessing language model deployment with risk cards.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1370–1380, Baltimore, Maryland. Association for Computational Linguistics.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H. Richemond, A. Doucet, Robin Strudel, Chris Dyer, Conor Durkan, Curtis Hawthorne, Rémi Leblond, Will Grathwohl, and Jonas Adler. 2022. Continuous diffusion for categorical data. *ArXiv*, abs/2211.15089.

Shuoyang Ding, Hainan Xu, and Philipp Koehn. 2019. Saliency-driven word alignment interpretation for neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation*.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

A. Seza Doğruöz, Sunayana Sitaram, Barbara E. Bullock, and Almeida Jacqueline Toribio. 2021. A survey of code-switching: Linguistic and social perspectives for language technologies. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1654–1666, Online. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

David M Eberhard, Gary F Simons, and Charles D. (eds.) Fennig. 2019. Ethnologue: Languages of the world. 2019. *online. Dallas, Texas: SIL International.*

Penelope Eckert. 2012. Three waves of variation study: The emergence of meaning in the study of sociolinguistic variation. *Annual review of Anthropology*, 41:87–100.

Penelope Eckert and Sally McConnell-Ginet. 2003. *Language and Gender*. Cambridge University Press.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 355–364, New Orleans, Louisiana. Association for Computational Linguistics.

Ahmed El-Kishky, Vishrav Chaudhary, Francisco Guzmán, and Philipp Koehn. 2020. CCAligned: A massive collection of cross-lingual web-document pairs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 5960–5969, Online. Association for Computational Linguistics.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21, Brussels, Belgium. Association for Computational Linguistics.

Aleksandr Ermolov, Aliaksandr Siarohin, E. Sangineto, and N. Sebe. 2020. Whitening for self-supervised representation learning. In *International Conference on Machine Learning*.

Miquel Esplà, Mikel Forcada, Gema Ramírez-Sánchez, and Hieu Hoang. 2019. ParaCrawl: Web-scale parallel corpora for the languages of the EU. In *Proceedings of Machine Translation Summit XVII: Translator, Project and User Tracks*, pages 118–119, Dublin, Ireland. European Association for Machine Translation.

Kawin Ethayarajh and Dan Jurafsky. 2022. The authenticity gap in human evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6056–6070, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Fahim Faisal, Sharlina Keshava, Md Mahfuz ibn Alam, and Antonios Anastasopoulos. 2021. SD-QA: Spoken Dialectal Question Answering for the Real World. Preprint.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proc. ACL*.

Shangbin Feng, Chan Young Park, Yuhan Liu, and Yulia Tsvetkov. 2023. From pretraining data to language models to downstream tasks: Tracking the trails of political biases leading to unfair NLP models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11737–11762, Toronto, Canada. Association for Computational Linguistics.

Acuña Ferreira. 2007. Language and Woman's Place. *Sociolinguistic Studies*.

Anjalie Field and Yulia Tsvetkov. 2020. Unsupervised discovery of implicit gender bias. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 596–608, Online. Association for Computational Linguistics.

John L. Fischer. 1958. Social influences on the choice of a linguistic variant. *WORD*.

Lucie Flek. 2020. Returning the N to NLP: Towards contextually personalized classification models. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, pages 7828–7838.

Aditya Gangrade, Anil Kag, and Venkatesh Saligrama. 2021. Selective classification via one-sided prediction. In *AISTATS*, pages 2179–2187.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*.

Xavier Garcia, Pierre Foret, Thibault Sellam, and Ankur Parikh. 2020. A multilingual view of unsupervised machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3160–3170, Online. Association for Computational Linguistics.

Xavier Garcia, Aditya Siddhant, Orhan Firat, and Ankur Parikh. 2021. Harnessing multilinguality in unsupervised machine translation for rare languages. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1126–1137, Online. Association for Computational Linguistics.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Saul B. Gelfand and Sanjoy K. Mitter. 1991. Recursive stochastic algorithms for global optimization in $\mathbb{R}^d$. *SIAM Journal on Control and Optimization*.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proc. EMNLP*.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for Twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association*

*for Computational Linguistics: Human Language Technologies*, pages 42–47, Portland, Oregon, USA. Association for Computational Linguistics.

Mark Girolami and Ben Calderhead. 2011. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*.

Gili Goldin, Ella Rabinovich, and Shuly Wintner. 2018. Native language identification with user generated content. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3591–3601, Brussels, Belgium. Association for Computational Linguistics.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM*, 63(11):139–144.

Sylviane Granger. 2003. The International Corpus of Learner English: a new resource for foreign language learning and teaching and second language acquisition research. *Tesol Quarterly*.

Alex Graves, Abdel rahman Mohamed, and Geoffrey E. Hinton. 2013. Speech recognition with deep recurrent neural networks. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649.

Robert M. Gray. 1990. Vector quantization. In *Readings in Speech Recognition*.

Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. 2019. Non-autoregressive neural machine translation with enhanced decoder input. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3723–3730.

Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. 2018. Dynamic task prioritization for multitask learning. In *Proc. ECCV*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6098–6111, Hong Kong, China. Association for Computational Linguistics.

Lelia Hampton. 2021. Black feminist musings on algorithmic oppression. *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*.

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 2023a. SSD-LM: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596, Toronto, Canada. Association for Computational Linguistics.

Xiaochuang Han, Sachin Kumar, Yulia Tsvetkov, and Marjan Ghazvininejad. 2023b. Ssd-2: Scaling and inference-time fusion of diffusion language models.

Xiaochuang Han, Daniel Simig, Todor Mihaylov, Yulia Tsvetkov, Asli Celikyilmaz, and Tianlu Wang. 2023c. Understanding in-context learning via supportive pretraining data. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12660–12673, Toronto, Canada. Association for Computational Linguistics.

Xiaochuang Han and Yulia Tsvetkov. 2021. Influence tuning: Demoting spurious correlations via instance attribution and instance-driven updates. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4398–4409, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaochuang Han and Yulia Tsvetkov. 2022. Orca: Interpreting prompted language models via locating supporting data evidence in the ocean of pretraining data.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021a. Decoupling adversarial training for fair NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 471–477, Online. Association for Computational Linguistics.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021b. Diverse adversaries for mitigating bias in training. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2760–2765, Online. Association for Computational Linguistics.

Moritz Hardt, Eric Price, and Nathan Srebro. 2016. Equality of opportunity in supervised learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3323–3331, Red Hook, NY, USA. Curran Associates Inc.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A probabilistic formulation of unsupervised text style transfer. In *International Conference on Learning Representations*.

Mark Heitmann, Christian Siebert, Jochen Hartmann, and Christina Schamp. 2020. More than a feeling: Benchmarks for sentiment analysis accuracy. *Available at SSRN 3489963*.

Daniel Hershcovich, Stella Frank, Heather Lent, Miryam de Lhoneux, Mostafa Abdou, Stephanie Brandl, Emanuele Bugliarello, Laura Cabello Piqueras, Ilias Chalkidis, Ruixiang Cui, Constanza Fierro, Katerina Margatina, Phillip Rust, and Anders Søgaard. 2022. Challenges and strategies in cross-cultural NLP. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6997–7013, Dublin, Ireland. Association for Computational Linguistics.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. 2022a. Video diffusion models. *arXiv:2204.03458*.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. 2022b. Video diffusion models. *ArXiv*, abs/2204.03458.

Cong Duy Vu Hoang, Gholamreza Haffari, and Trevor Cohn. 2017. Towards decoding as continuous optimisation in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 146–156, Copenhagen, Denmark. Association for Computational Linguistics.

Hieu Hoang and Philipp Koehn. 2008. Design of the Moses decoder for statistical machine translation. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 58–65, Columbus, Ohio. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Janet Holmes and Nick Wilson. 2017. *An Introduction to Sociolinguistics*. Routledge.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7038–7051, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*.

Juliane House and Gabriele Kasper. 2011. *Politeness Markers in English and German*, pages 157–186. De Gruyter Mouton, Berlin, New York.

Dirk Hovy. 2015a. Demographic factors improve classification performance. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (Volume 1: Long papers)*, pages 752–762.

Dirk Hovy. 2015b. Demographic factors improve classification performance. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 752–762, Beijing, China. Association for Computational Linguistics.

Dirk Hovy, Federico Bianchi, and Tommaso Fornaciari. 2020. "you sound just like your father" commercial machine translation systems include stylistic biases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1686–1690, Online. Association for Computational Linguistics.

Dirk Hovy and Shrimai Prabhumoye. 2021. Five sources of bias in natural language processing. *Language and Linguistics Compass*.

Dirk Hovy and Shannon L. Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598, Berlin, Germany. Association for Computational Linguistics.

Dirk Hovy and Diyi Yang. 2021. The importance of modeling social factors of language: Theory and practice. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 588–602, Online. Association for Computational Linguistics.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Mengting Hu, Yike Wu, Shiwan Zhao, Honglei Guo, Renhong Cheng, and Zhong Su. 2019. Domain-invariant feature distillation for cross-domain sentiment classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5559–5568, Hong Kong, China. Association for Computational Linguistics.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of International Conference on Machine Learning (ICML)*.

Xiaolei Huang and Michael J. Paul. 2019. Neural user factor adaptation for text classification: Learning to generalize across author demographics. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 136–146, Minneapolis, Minnesota. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Vivek Jayaram and John Thickstun. 2021. Parallel and flexible sampling from autoregressive models via langevin dynamics. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4807–4818. PMLR.

Monisha Jegadeesan, Sachin Kumar, John Wieting, and Yulia Tsvetkov. 2021. Improving the diversity of unsupervised paraphrasing with embedding outputs. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 166–175, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shihao Ji, S. V. N. Vishwanathan, Nadathur Satish, Michael J. Anderson, and Pradeep Dubey. 2016. Blackout: Speeding up recurrent neural network language models with very large vocabularies. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2022. Deep learning for text style transfer: A survey. *Computational Linguistics*, 48(1):155–205.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7:535–547.

Navya Jose, Bharathi Raja Chakravarthi, Shardul Suryawanshi, Elizabeth Sherly, and John P. McCrae. 2020. A survey of current datasets for code-switching research. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 136–141.

Aditya Joshi, Pushpak Bhattacharyya, Mark Carman, Jaya Saraswati, and Rajita Shukla. 2016. How do cultural differences impact the quality of sarcasm annotation?: A case study of Indian annotators and American text. In *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 95–99, Berlin, Germany. Association for Computational Linguistics.

Rafal Józefowicz, Oriol Vinyals, Mike Schuster, Noam M. Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *ArXiv*, abs/1602.02410.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proc. CVPR*.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019a. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019b. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Daniel Khashabi, Gabriel Stanovsky, Jonathan Bragg, Nicholas Lourie, Jungo Kasai, Yejin Choi, Noah A. Smith, and Daniel Weld. 2022. GENIE: Toward reproducible and standardized human evaluation for text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11444–11458, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Lidia Kidane, Sachin Kumar, and Yulia Tsvetkov. 2021. An exploration of data augmentation techniques for improving English to Tigrinya translation. In *Proceedings of the Second AfricaNLP Workshop*.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53, New Orleans, Louisiana. Association for Computational Linguistics.

Jyrki Kivinen and Manfred K. Warmuth. 1997. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1885–1894. JMLR.org.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2021. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*.

Moshe Koppel, Jonathan Schler, and Kfir Zigdon. 2005. Automatically determining an anonymous author's native language. *Intelligence and Security Informatics*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2020a. GeDi: Generative discriminator guided sequence generation.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2020b. GeDi: Generative Discriminator Guided Sequence Generation. *arXiv preprint arXiv:2009.06367*.

Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating unsupervised style transfer as paraphrase generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 737–762, Online. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Sachin Kumar, Antonios Anastasopoulos, Shuly Wintner, and Yulia Tsvetkov. 2021a. Machine translation into low-resource language varieties. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 110–121, Online. Association for Computational Linguistics.

Sachin Kumar, Vidhisha Balachandran, Lucille Njoo, Antonios Anastasopoulos, and Yulia Tsvetkov. 2023a. Language generation models can cause harm: So what can we do about it? an actionable survey. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3299–3321, Dubrovnik, Croatia. Association for Computational Linguistics.

Sachin Kumar, Soumen Chakrabarti, and Shourya Roy. 2017. Earth mover's distance pooling over siamese LSTMs for automatic short answer grading. In *Proc. IJCAI*.

Sachin Kumar, Eric Malmi, Aliaksei Severyn, and Yulia Tsvetkov. 2021b. Controlled text generation as continuous optimization with multiple constraints. In *Advances in Neural Information Processing Systems*.

Sachin Kumar, Biswajit Paria, and Yulia Tsvetkov. 2022a. Constrained sampling from language models via langevin dynamics in embedding spaces. *ArXiv*, abs/2205.12558.

Sachin Kumar, Biswajit Paria, and Yulia Tsvetkov. 2022b. Gradient-based constrained sampling from language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2251–2277, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sachin Kumar, Chan Young Park, and Yulia Tsvetkov. 2023b. A multivariate generative prompting framework with label descriptions for personalized zero-shot text classification.

Sachin Kumar and Yulia Tsvetkov. 2019. Von mises-fisher loss for training sequence to sequence models with continuous outputs. In *International Conference on Learning Representations*.

Sachin Kumar, Shuly Wintner, Noah A. Smith, and Yulia Tsvetkov. 2019a. Topics to avoid: Demoting latent confounds in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4153–4163, Hong Kong, China. Association for Computational Linguistics.

Sachin Kumar, Shuly Wintner, Noah A Smith, and Yulia Tsvetkov. 2019b. Topics to avoid: Demoting latent confounds in text classification. In *Proc. EMNLP*.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *Proceedings of International Conference on Machine Learning (ICML)*.

W. Labov. 1972. *Sociolinguistic Patterns*. University of Pennsylvania Press.

Surafel Melaku Lakew, Aliia Erofeeva, and Marcello Federico. 2018. Neural machine translation into language varieties. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 156–164, Brussels, Belgium. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018b. Word translation without parallel data. In *International Conference on Learning Representations*.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019a. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019b. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

Angeliki Lazaridou, Georgiana Dinu, and Marco Baroni. 2015. Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 270–280, Beijing, China. Association for Computational Linguistics.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proc. EMNLP*.

E.L. Lehmann and G. Casella. 1998. *Theory of Point Estimation*. Springer Verlag.

Klas Leino, Matt Fredrikson, Emily Black, Shayak Sen, and Anupam Datta. 2019. Feature-wise bias amplification. In *International Conference on Learning Representations*.

Erez Levon. 2007. Sexuality in context: Variation and the sociolinguistic perception of identity. *Language in Society*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland. Association for Computational Linguistics.

Mike Lewis and Angela Fan. 2019. Generative question answering: Learning to answer the whole question. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. 2022a. Diffusion-lm improves controllable text generation. *ArXiv*, abs/2205.14217.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. 2022b. Diffusion-lm improves controllable text generation. *ArXiv*, abs/2205.14217.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023. Unified demonstration retriever for in-context learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4644–4668, Toronto, Canada. Association for Computational Linguistics.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Xi Lin, Zhiyuan Yang, Qingfu Zhang, and Sam Kwong. 2021. Controllable pareto multi-task learning.

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. 2019. Pareto multi-task learning. In *Advances in Neural Information Processing Systems*.

Wang Ling, Chris Dyer, Alan W. Black, and Isabel Trancoso. 2015. Two/too simple adaptations of Word2Vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304, Denver, Colorado. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706, Online. Association for Computational Linguistics.

Guangyi Liu, Zichao Yang, Tianhua Tao, Xiaodan Liang, Junwei Bao, Zhen Li, Xiaodong He, Shuguang Cui, and Zhiting Hu. 2022. Don't take it literally: An edit-invariant sequence loss for text generation. In *Proc. NAACL*.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9).

Yijia Liu, Yi Zhu, Wanxiang Che, Bing Qin, Nathan Schneider, and Noah A. Smith. 2018. Parsing tweets into Universal Dependencies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 965–975, New Orleans, Louisiana. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach.

Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. 2018. Content preserving text generation with attribute controls. In *Advances in Neural Information Processing Systems*.

Yunfei Long, Qin Lu, Rong Xiang, Minglei Li, and Chu-Ren Huang. 2017. Fake news detection through multi-perspective speaker profiles. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 252–256, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Daming Lu. 2022. daminglu123 at SemEval-2022 task 2: Using BERT and LSTM to do text classification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 186–189, Seattle, United States. Association for Computational Linguistics.

Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, Noah A. Smith, and Yejin Choi. 2021a. Neurologic a*esque decoding: Constrained text generation with lookahead heuristics. *ArXiv*, abs/2112.08726.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021b. NeuroLogic decoding: (un)supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4288–4299, Online. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Eric Luhman and Troy Luhman. 2021. Knowledge distillation in iterative generative models for improved sampling speed.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Veronica Lynn, Youngseo Son, Vivek Kulkarni, Niranjan Balasubramanian, and H. Andrew Schwartz. 2017. Human centered NLP with user-factor adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1146–1155, Copenhagen, Denmark. Association for Computational Linguistics.

QING LYU, Marianna Apidianaki, and Chris Callison-Burch. 2022. Towards faithful model explanation in nlp: A survey. *ArXiv*, abs/2209.11326.

Qingsong Ma, Johnny Wei, Ondřej Bojar, and Yvette Graham. 2019. Results of the WMT19 metrics shared task: Segment-level and strong MT systems pose big challenges. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 62–90, Florence, Italy. Association for Computational Linguistics.

Shervin Malmasi and Mark Dras. 2018. Native language identification with classifier stacking and ensembles. *Computational Linguistics*, 44(3):403–446.

Shervin Malmasi, Keelan Evanini, Aoife Cahill, Joel Tetreault, Robert Pugh, Christopher Hamill, Diane Napolitano, and Yao Qian. 2017. A report on the 2017 native language identification shared task. In *Workshop on Innovative Use of NLP for Building Educational Applications*.

P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65.

Kelly Marchisio, Kevin Duh, and Philipp Koehn. 2020. When does unsupervised machine translation work? In *Proceedings of the Fifth Conference on Machine Translation*, pages 571–583, Online. Association for Computational Linguistics.

Luis Marujo, Nuno Grazina, Tiago Luis, Wang Ling, Luisa Coheur, and Isabel Trancoso. 2011. BP2EP - adaptation of Brazilian Portuguese texts to European Portuguese. In *Proceedings of the 15th Annual Conference of the European Association for Machine Translation*, Leuven, Belgium. European Association for Machine Translation.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Jon D. McAuliffe and David M. Blei. 2008. Supervised topic models. In *NeurIPS*.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *ACM Comput. Surv.*, 54(6).

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2022. Typical decoding for natural language generation.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2023. Locally typical sampling. *Transactions of the Association for Computational Linguistics*, 11:102–121.

Lars M. Mescheder, Andreas Geiger, and Sebastian Nowozin. 2018. Which training methods for gans do actually converge? In *International Conference on Machine Learning*.

Miriam Meyerhoff and Susan Ehrlich. 2019. Language, gender, and sexuality. *Annual Review of Linguistics*.

Paul Michel, Tatsunori Hashimoto, and Graham Neubig. 2021. Modeling the second player in distributionally robust optimization. In *International Conference on Learning Representations*.

Paul Michel and Graham Neubig. 2018. MTNT: A testbed for machine translation of noisy text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 543–553, Brussels, Belgium. Association for Computational Linguistics.

Sabrina J. Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, and Samson Tan. 2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2021. Noisy channel language model prompting for few-shot text classification. *arXiv preprint*.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Noisy channel language model prompting for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5316–5330, Dublin, Ireland. Association for Computational Linguistics.

Fatemehsadat Mireshghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. 2022a. Mix and match: Learning-free controllable text generation using energy language models.

Fatemehsadat Mireshghallah, Vaishnavi Shrivastava, Milad Shokouhi, Taylor Berg-Kirkpatrick, Robert Sim, and Dimitrios Dimitriadis. 2022b. UserIdentifier: Implicit user representations for simple and effective personalized sentiment analysis. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3449–3456, Seattle, United States. Association for Computational Linguistics.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 2265–2273, Red Hook, NY, USA. Curran Associates Inc.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature*.

Ioannis Mollas, Zoe Chrysopoulou, Stamatis Karlos, and Grigorios Tsoumakas. 2022. ETHOS: a multi-label hate speech detection dataset. *Complex & Intelligent Systems*.

Burt L. Monroe, Michael P. Colaresi, and Kevin M. Quinn. 2008. Fightin' words: Lexical feature selection and evaluation for identifying the content of political conflict. *Political Analysis*.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *International Conference on Artificial Intelligence and Statistics*.

Preslav Nakov and Jörg Tiedemann. 2012. Combining word-level and character-level models for machine translation between closely-related languages. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 301–305, Jeju Island, Korea. Association for Computational Linguistics.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6341–6350. Curran Associates, Inc.

Lucille Njoo, Chan Young Park, Octavia Stappart, Marvin Thielk, Yi Chu, and Yulia Tsvetkov. 2023. Talkup: A novel dataset paving the way for understanding empowering language.

Eva Ogiermann. 2009. Politeness and in-directness across cultures: A comparison of english, german, polish and russian requests. *Journal of Politeness Research*, 5(2):189–216.

Joseph O'Neill, Barty Pleydell-Bouverie, David Dupret, and Jozsef Csicsvari. 2010. Play it again: reactivation of waking experience and memory. *Trends in Neurosciences*.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online. Association for Computational Linguistics.

Alissa Ostapenko, Shuly Wintner, Melinda Fricke, and Yulia Tsvetkov. 2022. Speaker information can guide models to better inductive biases: A case study on predicting code-switching. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3853–3867, Dublin, Ireland. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Nikolaos Pappas, Phoebe Mulcaire, and Noah A. Smith. 2020. Grounded compositional outputs for adaptive language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1252–1267, Online. Association for Computational Linguistics.

Biswajit Paria, Chih-Kuan Yeh, Ian E. H. Yen, Ning Xu, Pradeep Ravikumar, and Barnabás Póczos. 2020. Minimizing FLOPs to learn efficient sparse representations. In *International Conference on Learning Representations*.

Chan Young Park and Yulia Tsvetkov. 2019. Learning to generate word- and phrase-embeddings for efficient phrase-based neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 241–248, Hong Kong. Association for Computational Linguistics.

Damian Pascual, Beni Egressy, Clara Meister, Ryan Cotterell, and Roger Wattenhofer. 2021. A plug-and-play method for controlled text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3973–3997, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sam Patterson and Yee Whye Teh. 2013. Stochastic gradient riemannian langevin dynamics on the probability simplex. In *NuerIPS*.

Michael J. Paul. 2017. Feature selection as causal inference: Experiments with text classification. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 163–172, Vancouver, Canada. Association for Computational Linguistics.

Judea Pearl. 2009. *Causality: Models, Reasoning and Inference*. Cambridge University Press.

Jiaxin Pei and David Jurgens. 2023. When do annotator demographics matter? measuring the influence of annotator demographics with the popquorn dataset.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

John Platt and Alan Barr. 1988. Constrained differential optimization. In *Advances in Neural Information Processing Systems*. American Institute of Physics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Nima Pourdamghani and Kevin Knight. 2017. Deciphering related languages. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2513–2518, Copenhagen, Denmark. Association for Computational Linguistics.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876, Melbourne, Australia. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2020. Learning to deceive with attention-based explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4782–4793, Online. Association for Computational Linguistics.

Reid Pryzant, Kelly Shen, Dan Jurafsky, and Stefan Wagner. 2018. Deconfounded lexicon induction for interpretable social science. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1615–1625, New Orleans, Louisiana. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu Chen. 2022. Controllable natural language generation with contrastive prefixes. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2912–2924, Dublin, Ireland. Association for Computational Linguistics.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D. Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. 2020. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 794–805, Online. Association for Computational Linguistics.

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. 2022. COLD decoding: Energy-based constrained text generation with langevin dynamics.

Qu Qu, Zheng Ma, Anders Clausen, and Bo Nørregaard Jørgensen. 2021. A comprehensive review of machine learning in multi-objective optimization. In *2021 IEEE 4th International Conference on Big Data and Artificial Intelligence (BDAI)*, pages 7–14.

Ella Rabinovich, Yulia Tsvetkov, and Shuly Wintner. 2018. Native language cognate effects on second language lexical choice. *Transactions of the Association for Computational Linguistics*, 6:329–342.

Bahar Radfar, Karthik Shivaram, and Aron Culotta. 2020. Characterizing variation in toxic language by social context. *Proceedings of the International AAAI Conference on Web and Social Media*, 14(1):959–963.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019a. Language models are unsupervised multitask learners.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019b. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Rajat Raina, Yirong Shen, Andrew McCallum, and Andrew Ng. 2003. Classification with hybrid generative/discriminative models. In *Advances in Neural Information Processing Systems*, volume 16. MIT Press.

Dheeraj Rajagopal, Vidhisha Balachandran, Eduard H Hovy, and Yulia Tsvetkov. 2021. SELFEXPLAIN: A self-explaining architecture for neural text classifiers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 836–850, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140, New Orleans, Louisiana. Association for Computational Linguistics.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. *arXiv preprint arXiv:2004.07667*.

Shauli Ravfogel, Michael Twiton, Yoav Goldberg, and Ryan D Cotterell. 2022. Linear adversarial concept erasure. In *International Conference on Machine Learning*, pages 18400–18421. PMLR.

John Rawls. 1999. *A Theory of Justice*. Harvard University Press.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020a. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020b. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Virgile Landeiro Dos Reis and Aron Culotta. 2018. Robust text classification under confounding shift. *Journal of Artificial Intelligence Research*.

Dor Ringel, Gal Lavee, Ido Guy, and Kira Radinsky. 2019. Cross-cultural transfer learning for text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3873–3883, Hong Kong, China. Association for Computational Linguistics.

Paul R. Rosenbaum and Donald B. Rubin. 1985. Constructing a control group using multivariate matched sampling methods that incorporate the propensity score. *The American Statistician*.

Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. 2017. Stabilizing training of generative adversarial networks through regularization. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 2015–2025, Red Hook, NY, USA. Curran Associates Inc.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *J. Artif. Int. Res.*, 65(1):569–630.

Diego Ruiz-Antolín and Javier Segura. 2016. A new type of sharp bounds for ratios of modified bessel functions. *Journal of Mathematical Analysis and Applications*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Maarten Sap, Ronan Le Bras, Daniel Fried, and Yejin Choi. 2022a. Neural theory-of-mind? on the limits of social intelligence in large LMs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3762–3780, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Maarten Sap, Swabha Swayamdipta, Laura Vianna, Xuhui Zhou, Yejin Choi, and Noah A. Smith. 2022b. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5884–5906, Seattle, United States. Association for Computational Linguistics.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Teven Le Scao, Angela Fan, Christopher Akiki, Elizabeth-Jane Pavlick, Suzana Ili'c, Daniel Hesslow, Roman Castagn'e, Alexandra Sasha Luccioni, Franccois Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Rose Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurenccon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa Etxabe, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris C. Emezue, Christopher Klamm, Colin Leong, Daniel Alexander van Strien, David Ifeoluwa Adelani, Dragomir R. Radev, Eduardo Gonz'alez Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady ElSahar, Hamza Benyamina, Hieu Trung Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jorg Frohberg, Josephine L. Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro von Werra, Leon Weber, Long Phan, Loubna Ben Allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, Mar'ia Grandury, Mario vSavsko, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad Ali Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla A. Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto L'opez, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, S. Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal V. Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Févry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiang Tang, Zheng Xin Yong, Zhiqing Sun, Shaked Brody, Y Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre Franccois Lavall'ee, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aur'elie N'ev'eol, Charles Lovering, Daniel H Garrette, Deepak R. Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Xiangru Tang, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, S. Osher Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav

Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdenvek Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ananda Santa Rosa Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Olusola Ajibade, Bharat Kumar Saxena, Carlos Muñoz Ferrandis, Danish Contractor, David M. Lansky, Davis David, Douwe Kiela, Duong Anh Nguyen, Edward Tan, Emily Baylor, Ezinwanne Ozoani, Fatim Tahirah Mirza, Frankline Ononiwu, Habib Rezanejad, H.A. Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jan Passmore, Joshua Seltzer, Julio Bonis Sanz, Karen Fort, Lívia Macedo Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, M. K. K. Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nourhan Fahmy, Olanrewaju Samuel, Ran An, R. P. Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas L. Wang, Sourav Roy, Sylvain Viguier, Thanh-Cong Le, Tobi Oyebade, Trieu Nguyen Hai Le, Yoyo Yang, Zachary Kyle Nguyen, Abhinav Ramesh Kashyap, A. Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Kumar Singh, Benjamin Beilharz, Bo Wang, Caio Matheus Fonseca de Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel Le'on Perin'an, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Iman I.B. Bello, Isha Dash, Ji Soo Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthi Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, María Andrea Castillo, Marianna Nezhurina, Mario Sanger, Matthias Samwald, Michael Cullan, Michael Weinberg, M Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patricia Haller, R. Chandrasekhar, R. Eisenberg, Robert Martin, Rodrigo L. Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Pratap Bharati, T. A. Laud, Th'eo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yashasvi Bajaj, Y. Venkatraman, Yifan Xu, Ying Xu, Yu Xu, Zhee Xao Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. 2022. Bloom: A 176b-parameter open-access multilingual language model. *ArXiv*, abs/2211.05100.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with

subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Judy Hanwen Shen, Lauren Fratamico, Iyad Rahwan, and Alexander M. Rush. 2018. Darling or babygirl? investigating stylistic bias in sentiment analysis.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, Hong Kong, China. Association for Computational Linguistics.

Emily Sheng and David Uthus. 2020. Investigating societal biases in a poetry composition system.

Weijia Shi, Xiaochuang Han, Hila Gonen, Ari Holtzman, Yulia Tsvetkov, and Luke Zettlemoyer. 2022. Toward human readable prompt tuning: Kubrick's the shining is a good movie, and a good prompt too?

Chenglei Si, Dan Friedman, Nitish Joshi, Shi Feng, Danqi Chen, and He He. 2023a. Measuring inductive biases of in-context learning with underspecified demonstrations. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11289–11310, Toronto, Canada. Association for Computational Linguistics.

Chenglei Si, Dan Friedman, Nitish Joshi, Shi Feng, Danqi Chen, and He He. 2023b. What spurious features can pretrained language models combat?

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of International Conference on Machine Learning (ICML)*.

Congzheng Song, Alexander Rush, and Vitaly Shmatikov. 2020. Adversarial semantic collisions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4198–4210, Online. Association for Computational Linguistics.

Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*.

Till Speicher, Hoda Heidari, Nina Grgic-Hlaca, Krishna P Gummadi, Adish Singla, Adrian Weller, and Muhammad Bilal Zafar. 2018. A unified approach to quantifying algorithmic unfairness: Measuring individual & group unfairness via inequality indices. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2239–2248.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

Robin Strudel, Corentin Tallec, Florent Altché, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, and Rémi Leblond. 2022. Self-conditioned embedding diffusion for text generation.

Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 2019. "transforming" delete, retrieve, generate approach for controlled text style transfer. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3269–3279, Hong Kong, China. Association for Computational Linguistics.

Jimin Sun, Hwijeen Ahn, Chan Young Park, Yulia Tsvetkov, and David R. Mortensen. 2021. Cross-cultural similarity features for cross-lingual transfer learning of pragmatically motivated tasks. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2403–2414, Online. Association for Computational Linguistics.

Jiuding Sun, Chantal Shaib, and Byron C. Wallace. 2023. Evaluating the zero-shot robustness of instruction-tuned language models.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Chenhao Tan, Lillian Lee, and Bo Pang. 2014. The effect of wording on message propagation: Topic- and author-controlled natural experiments on Twitter. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 175–185, Baltimore, Maryland. Association for Computational Linguistics.

T. Tan, S. Goh, and Y. Khaw. 2012. A malay dialect translation and synthesis system: Proposal and preliminary system. In *2012 International Conference on Asian Language Processing*, pages 109–112.

Ishan Tarunesh, Syamantak Kumar, and Preethi Jyothi. 2021. From machine translation to code-switching: Generating high-quality code-switched text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3154–3169, Online. Association for Computational Linguistics.

Rachael Tatman. 2017. Gender and dialect bias in YouTube's automatic captions. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 53–59, Valencia, Spain. Association for Computational Linguistics.

Yi Tay, Vinh Q. Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. 2022. Charformer: Fast character transformers via gradient-based subword tokenization. In *International Conference on Learning Representations*.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. In *Workshop on Building Educational Applications Using NLP*.

Jörg Tiedemann. 2009. Character-based PSMT for closely related languages. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation*, Barcelona, Spain. European Association for Machine Translation.

Alexandru Tifrea, Gary Becigneul, and Octavian-Eugen Ganea. 2019. Poincare glove: Hyperbolic word embeddings. In *International Conference on Learning Representations*.

Oren Tsur and Ari Rappoport. 2007. Using classifier features for studying the effect of native language on the choice of written second language words. In *Workshop on Cognitive Aspects of Computational Language Acquisition*.

Miles Turpin, Julian Michael, Ethan Perez, and Sam Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *ArXiv*, abs/2305.04388.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. Deep domain confusion: Maximizing for domain invariance. ArXiv:1412.3474.

Eva Vanmassenhove, Dimitar Shterionov, and Andy Way. 2019. Lost in translation: Loss and decay of linguistic richness in machine translation. In *Proceedings of Machine Translation Summit XVII: Research Track*, pages 222–232, Dublin, Ireland. European Association for Machine Translation.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. In *Advances in Neural Information Processing Systems*.

David Vilar, Jan-T. Peter, and Hermann Ney. 2007. Can we translate letters? In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, page 33–39, USA. Association for Computational Linguistics.

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *ArXiv*, abs/1506.05869.

Svitlana Volkova, Theresa Wilson, and David Yarowsky. 2013a. Exploring demographic language variations to improve multilingual sentiment analysis in social media. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1815–1827, Seattle, Washington, USA. Association for Computational Linguistics.

Svitlana Volkova, Theresa Wilson, and David Yarowsky. 2013b. Exploring sentiment in social media: Bootstrapping subjectivity clues from multilingual twitter streams. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 505–510.

Minghan Wang, Guo Jiaxin, Yuxia Wang, Yimeng Chen, Su Chang, Hengchao Shang, Min Zhang, Shimin Tao, and Hao Yang. 2021a. How length prediction influence the performance of non-autoregressive translation? In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 205–213, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Weichao Wang, Shi Feng, Wei Gao, Daling Wang, and Yifei Zhang. 2018. Personalized microblog sentiment classification via adversarial cross-lingual multi-task learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Brussels, Belgium. Association for Computational Linguistics.

Xinyi Wang, Sebastian Ruder, and Graham Neubig. 2021b. Multi-view subword regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 473–482, Online. Association for Computational Linguistics.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. 2023. Large language models are implicitly topic models: Explaining and finding good demonstrations for in-context learning.

Zijian Wang and Christopher Potts. 2019. TalkDown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3711–3719, Hong Kong, China. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022a. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, Courtney Biles, Sasha Brown, Zac Kenton, Will Hawkins, Tom Stepleton, Abeba Birhane, Lisa Anne Hendricks, Laura Rimell, William Isaac, Julia Haas, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2022a. Taxonomy of risks posed by language models. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 214–229, New York, NY, USA. Association for Computing Machinery.

Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, et al. 2022b. Taxonomy of risks posed by language models. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 214–229.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

Max Welling and Yee Whye Teh. 2011. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of International Conference on Machine Learning (ICML)*.

Sarah Wiegreffe, Ana Marasović, and Noah A. Smith. 2021. Measuring association between labels and free-text rationales. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10266–10284, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Anna Wierzbicka. 2020. *Cross-Cultural Pragmatics*. De Gruyter Mouton, Berlin, Boston.

John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. 2019. Beyond BLEU:training neural machine translation with semantic similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4344–4355, Florence, Italy. Association for Computational Linguistics.

Gian Wiher, Clara Meister, and Ryan Cotterell. 2022. On Decoding Strategies for Neural Text Generators. *Transactions of the Association for Computational Linguistics*, 10:997–1012.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew L Williams, Pete Burnap, and Luke Sloan. 2017. Towards an ethical framework for publishing twitter data in social research: Taking into account users' views, online context and algorithmic estimation. *Sociology*, 51(6):1149–1168. PMID: 29276313.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Sze-Meng Jojo Wong and Mark Dras. 2009. Contrastive analysis and native language identification. In *Proc. Australasian Language Technology Association Workshop*.

Sze-Meng Jojo Wong and Mark Dras. 2011. Exploiting parse structures for native language identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1600–1610, Edinburgh, Scotland, UK. Association for Computational Linguistics.

David Wright. 2013. Stylistic variation within genre conventions in the enron email corpus: developing a textsensitive methodology for authorship research. *International Journal of Speech, Language and the Law*, 20(1):45–75.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing

Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Mengzhou Xia, Anjalie Field, and Yulia Tsvetkov. 2020. Demoting racial bias in hate speech detection. In *Proceedings of the Eighth International Workshop on Natural Language Processing for Social Media*, pages 7–14, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. 2017. Controllable invariance through adversarial feature learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 585–596, Red Hook, NY, USA. Curran Associates Inc.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, Denver, Colorado. Association for Computational Linguistics.

Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Michael L Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. 2017. Toward human parity in conversational speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(12):2410–2423.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530.

Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, Online. Association for Computational Linguistics.

Kexin Yang, Dayiheng Liu, Wenqiang Lei, Baosong Yang, Mingfeng Xue, Boxing Chen, and Jun Xie. 2023. Tailor: A soft-prompt-based approach to attribute-based controlled text generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 410–427, Toronto, Canada. Association for Computational Linguistics.

Yi Yang and Jacob Eisenstein. 2017. Overcoming language variation in sentiment analysis with social attention. *Transactions of the Association for Computational Linguistics*, 5:295–307.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Kyra Yee, Yann Dauphin, and Michael Auli. 2019. Simple and effective noisy channel modeling for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5696–5701.

Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. 2016. Neural generative question answering. In *Proceedings of the Workshop on Human-Computer Question Answering*, pages 36–42, San Diego, California. Association for Computational Linguistics.

D Yogatama, C Dyer, W Ling, and P Blunsom. 2017a. Generative and discriminative text classification with recurrent neural networks. In *Thirty-fourth International Conference on Machine Learning (ICML 2017)*. International Machine Learning Society.

Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. 2017b. Generative and discriminative text classification with recurrent neural networks.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proc. AAAI*.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. 2021. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTscore: Evaluating text generation with BERT. In *International Conference on Learning Representations*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*.

He Zhao, Dinh Q. Phung, Viet Huynh, Yuan Jin, Lan Du, and Wray L. Buntine. 2021a. Topic modelling meets deep neural networks: A survey. In *International Joint Conference on Artificial Intelligence*.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021b. Calibrate before use: Improving few-shot performance of language models.

Chunting Zhou, Xuezhe Ma, Paul Michel, and Graham Neubig. 2021. Examining and combating spurious features under distribution shift. In *International Conference on Machine Learning (ICML)*, Virtual.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences.

Caleb Ziems, William Held, Jingfeng Yang, Jwala Dhamala, Rahul Gupta, and Diyi Yang. 2023. Multi-VALUE: A framework for cross-dialectal English NLP. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 744–768, Toronto, Canada. Association for Computational Linguistics.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The United Nations parallel corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, Portorož, Slovenia. European Language Resources Association (ELRA).

George Kingsley Zipf. 1935. The psycho-biology of language.