Crosslingual Sharing for Low-Resource Natural Language Processing

Phoebe Mulcaire

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2022

Reading Committee:

Noah A. Smith, Chair

Luke Zettlemoyer

Richard Anderson

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Crosslingual Sharing for
Low-Resource Natural Language Processing

Phoebe Mulcaire

Chair of the Supervisory Committee:
Professor Noah A. Smith
Computer Science and Engineering

Modern NLP systems have been highly successful at a wide variety of tasks, including language modeling and structured prediction problems such as syntactic and semantic parsing. This is due in large part to the use of supervised neural networks and more recently to unsupervised contextualized representations. However, these techniques rely on resources, such as extensive task-specific annotation and vast amounts of unlabeled text, which are not available in every language. Thus, most prior research has been focused on high-resource languages such as English. Crosslingual transfer from a high-resource source language, or sharing among many languages, has increasingly been used to achieve similar improvements in low-resource languages by exploiting underlying similarities between languages; however, many techniques for crosslingual transfer in turn require crosslingual resources such as parallel corpora, which again may not be available. All of these factors pose challenges to natural language processing in low-resource languages.

This thesis argues that even with little, indirect or absent crosslingual supervision, shar-

ing information between languages is a highly effective strategy for low-resource NLP, and quantifies the benefits in various low-resource settings and languages. We describe two lines of work addressing the problem of crosslingual transfer in such low-resource settings. In the first, we present language models and supervised structured prediction models which take a joint training approach, sharing parameters across several languages, to improve performance relative to monolingual training. We begin the second with GroC, a language model with compositional input and output representations which store linguistic information independently of any specific vocabulary, and show that GroC succeeds in low-resource language modeling and monolingual domain adaptation. Finally, we unite these two threads by using joint crosslingual training for compositional language models, including ones which use crosslingual lexicons not available to previous multilingual models. We show that this combined approach improves low-resource learning for a variety of target languages.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This thesis would never have been possible if not for the help and support of many people, for which I am deeply grateful.

I want to thank first of all my advisor, Noah Smith, for his support, mentorship, and kindness, and for all our discussions that went into the research presented here. The research I have done over the past seven years is what it is thanks to Noah's encouragement to have high standards for my work. Thanks also to my other committee members—Luke Zettlemoyer, Richard Anderson, and Fei Xia—for their thoughtful questions and feedback.

Enormous amounts of praise are due to my coauthors Swabha Swayamdipta, Jungo Kasai, and Nikolaos Pappas, who made major contributions of time and effort to these research projects; thank you all very much for working together with me. Special thanks to Waleed Ammar, for mentoring me in my first year, and helping set me on this path.

I'm grateful to all of Noah's ARK, and the UWNLP community as a whole, for friendly and clever conversations; in addition to my coauthors, I want to thank Sofia Serrano, Hila Gonen, Suchin Gururangan, Jesse Dodge, Julian Michael, Maxwell Forbes, Ari Holtzman, Hao Peng, Eunsol Choi, and Dallas Card. And especially, thanks to Maarten Sap, Lucy Lin, Elizabeth Clark, and Kelvin Luu, for being there from the beginning, and for lunches together.

Many thanks to my family, who patiently listened to my fretting about each paper, the job search and eventually my thesis writing.

And finally, thanks to Taylor Friesen, for support, kindness, thoughtful questions, time, effort, friendship, cleverness, lunches, patience, love, and everything else.

Chapter 1

INTRODUCTION

This thesis considers the problem of natural language processing (NLP) in low-resource settings, and addresses it by means of sharing information between languages. It contributes experimental evidence that at small scales, neural models for NLP can benefit from the use of multilingual data, as well as architectures designed to make efficient use of limited data. These facts have implications for the extension of NLP models, which are currently highly successful in English and certain other high-resource languages, to languages with less available data.

What exactly it means for a computer to learn a language is not a settled question, but analyzing the structure of a sentence, translating from one language to another, or summarizing a passage are all tasks which modern machine learning models regularly undertake, and to a great extent succeed at—in some languages, at least. But whether we wish to provide practical tools to speakers of a language or to understand the way that language behaves as a matter of scientific inquiry, it is worthwhile to pursue these goals for *every human language*, not just a few. Currently, though, there are thousands of living human languages for which these goals are not met by existing models. To clarify the significance of this problem, we first place it in historical context.

1.1 Background: A Brief History of Multilingual NLP

Since the earliest formulations of natural language processing (Weaver, 1952; McCarthy et al., 1955), researchers have sought linguistic capacities more general than any one language—translating between languages by understanding the shared semantics underlying two sentences, for example (see Chapter 2 for work on multilingual semantics), or defining new words in terms of ones already known (see Chapter 4). Historically, though, rule-based approaches meant that models of natural language depended on developers' knowledge of the language, with the consequence that research efforts, even when using in-principle language-independent theories, tended to cluster in a few well-studied languages (Bender et al., 2002; Beesley and Karttunen, 2003). In the late 1980s and 1990s, attention shifted to "empiricist" or statistical methods based on analyzing large quantities of language data (Church and Mercer, 1993). Some researchers saw in this an opportunity from multilinguality: any language could be handled statistically, given data in that language (Cucerzan and Yarowsky, 2000; Diab and Resnik, 2002; Dumais et al., 1997, inter alia). But while many tasks could be framed in language-independent ways, varying amounts of resources meant that using the same techniques for different languages met with varying amounts of suc-

cess (Resnik et al., 1999). A variety of multilingual resource-boosting solutions flourished for a while: annotation projection via parallel corpora (Yarowsky et al., 2001; Hwa et al., 2005) or machine translation of labeled data to a target language (Durrett et al., 2012; Duh et al., 2011) enabled statistical NLP techniques for lower-resource languages by combining annotation in high-resource languages with crosslingual supervision. Model transfer (Zeman and Resnik, 2008) approaches used delexicalized data, annotated with language-universal features such as part-of-speech tags, to learn a language-general model on source language data or a combination of source and target data, then apply it to the target language.

With the rise of neural networks and continuous vector representations in the 2010s (Mikolov et al., 2013; Faruqui and Dyer, 2014), a *polyglot* approach to NLP (Tsvetkov et al., 2016) gained popularity: models that were trained on and applied to multiple languages at once, generalizing over the variation between languages (Täckström et al., 2012; Guo et al., 2015; Ammar et al., 2016a,b). Multilingual word vectors enabled the representation of words from multiple languages in a shared semantic vector space, and parameter sharing in neural networks allowed hidden representations to be similarly multilingual. This approach has persisted through subsequent shifts in the field, such as replacing word vectors with contextualized representations (Chapter 3) and replacing task-specific models with large language models in the pretraining-finetuning paradigm (Devlin et al., 2019). It has now arguably become dominant in the field: in the past few years, state of the art large language models, whether they are focused on multilinguality (Xue et al., 2021) or not (Chowdhery et al., 2022), are routinely evaluated for multilingual capabilities developed from the inclusion of multilingual training text, with little language-specific design and no crosslingual supervision.

1.2 This Work

Polyglot models are preferred for their ease of use and for their potential to solve the problem of low-resource languages, where sufficient data for neural methods is lacking: by sharing information between languages, high-resource languages can improve the performance of low-resource ones. This is especially critical as scale increases, because resource disparities in existing datasets are stark. Common Crawl, a repository of content scraped from the web which is frequently used as a source of text data, includes text from hundreds of languages—but more than 45% of the documents crawled are in English, just ten languages together account for over 82%, and Hindi, the world's third most-spoken language, represents less than 0.13%.[1]

However, crosslingual sharing at scale is not, by itself, a complete answer to the problem of multilingual NLP. Results in this thesis suggest, and other contemporary work (Pires

[1]Statistics based on the CC-MAIN-2022-05 crawl. See https://commoncrawl.github.io/cc-crawl-statistics/plots/languages.

et al., 2019; Singh et al., 2019; Lauscher et al., 2020) corroborates, that crosslingual transfer through polyglot models is more successful for typologically or phylogenetically close languages. If typological similarity or phylogenetic relatedness is important for successful crosslingual transfer, then many languages (isolates, or languages whose nearest relatives are all low-resource) may still be left behind. The "curse of multilinguality", or the tendency of crosslingual performance to degrade as more languages are added to a model (Conneau et al., 2020), also increases the requirements for massively multilingual models' capacity, which can be extremely burdensome at large scales.

1.3 Roadmap

In this thesis, we explore approaches to the prediction of linguistic structures and language modeling based on the principle of crosslingual sharing, with a focus on very low-resource scenarios. To this end, we consider tasks (named entity recognition, semantic role labeling, and dependency parsing) that involve predicting linguistic structures or labels for a given sentence. When evaluating language models, we focus on perplexity, again for the ability to directly quantify relatively subtle changes in representation quality for classes of words, rather than via a downstream proxy such as question answering. While state of the art models are now frequently evaluated via natural language generation in response to prompts or templates, we choose evaluations that allow relatively fine-grained, quantifiable analysis of the types of information that are successfully transferred, finding for example that common labels benefit more from polyglot training than rare ones. These evaluations provide more meaningful information than can be obtained from generation tasks, as this work studies the ability of very-low-resource models to learn low-level syntax and semantics, rather than pragmatics, world knowledge, and common sense.

We investigate the mechanisms of polyglot training by exploring its effect at small scales, and quantify the benefits to low-resource NLP in a variety of settings and languages. In the first part of this work, Polyglot Modeling, we describe two models that address the low-resource problem through joint multilingual training, in which large numbers of parameters or a full model are trained on data from multiple languages. Our hypothesis is that, although each language is unique, different languages manifest similar characteristics (e.g., morphological, lexical, syntactic) which can be exploited by training a single model with data from multiple languages (Ammar, 2016).

Inspired by the approach of Ammar et al. (2016a), we apply this idea—which we call polyglot training—to PropBank-style semantic role labeling (SRL) in Chapter 2. We train several parsers for each language in the CoNLL 2009 dataset (Hajič et al., 2009): a traditional monolingual version, and variants which additionally incorporate supervision from the English portion of the dataset. To our knowledge, this is the first multilingual SRL approach to directly combine supervision from multiple languages. We find that the polyglot approach outperforms the monolingual approach on most languages, and that the

improvement is greater the less data the target language has. This is true even when the annotation labels of the target language are different from those of English, suggesting that the model learns similarities between related labels with different names. We find that even a simple combination of data is as effective as more complex kinds of polyglot training, and analyze specific label accuracies to show that transfer is most effective for more common labels in a given language.

In Chapter 3 we describe Rosita, a jointly trained multilingual language model for producing multilingual contextualized representations. With this model, we explore crosslingual transfer between highly dissimilar languages (English→Chinese and English→Arabic) for three core tasks: SRL, named entity recognition (NER) and Universal Dependency (UD) parsing. Our experiments cover comparisons in three dimensions: monolingual vs. polyglot representations, contextual vs. word type embeddings, and, within the contextual representation paradigm, purely character-based language models vs. ones that include word-level input. We show that for most language-task combinations, polyglot models can improve over monolingual ones, even in a relatively high-resource setting. Sections 3.7 and 3.8 provide comparisons to a different method of producing multilingual contextualized word representations, and demonstrate that joint crosslingual training performs better with fewer requirements. We also experiment with few-shot and zero-shot transfer for a diverse set of languages, showing that joint crosslingual training produces reliable improvements for low-resource dependency parsing.

In the second part of the thesis, Vocabulary Independence, we discuss another approach to low-resource modeling, first examining it in a monolingual setting and then extending it to crosslingual modeling. GroC, short for Grounded Output Composition, is an approach to word representations for language modeling that combines features describing surface character sequences, semantic relations, and word definitions from a lexical resource like WordNet (Fellbaum, 1998). This parameterization means that GroC can assign probability to words not seen during training, allowing a vocabulary different from the training vocabulary—e.g., one associated with a different text domain or language—to be considered at inference time. It also allows the model to share information between words if they have similar spellings, semantic relations, or definitions. In Chapter 4 we show experimentally that this property improves perplexity in a variety of language modeling settings, and that the perplexity gains are strongest for low-frequency words, implying improved sample efficiency relative to baselines: compositional output representations allow us to predict words from fewer training examples. In particular, we demonstrate improved performance for low-resource and zero-resource cross-domain transfer, in which a language model is trained on one domain and then evaluated on another domain with a small amount of target domain finetuning or without any finetuning, respectively. While the ability to adapt to new domains is by itself useful for low-resource language modeling, it also hints at potential for *crosslingual* adaptation, which can be seen as an extreme sort of domain adaptation (Prettenhofer and Stein, 2011).

Finally, following GroC's idea of decoupling model and vocabulary, we blend "polyglot modeling" and "vocabulary independence" in Chapter 5 by experimenting with multilingual extensions to GroC, including multilingual joint training of compositional embeddings and incorporation of crosslingual lexicons for multilingual grounding. We compare several kinds of models to examine the effect of partially or fully compositional word representations in several languages. We find that polyglot modeling and vocabulary independence are compatible, and when applied together improve low-resource learning beyond either one alone. We also find that crosslingual training of compositional embedding networks produces robustly alignable embedding spaces that can be extended to out-of-vocabulary words thanks to the property of vocabulary independence.

Together, this research contributes to a better understanding of the mechanisms of crosslingual sharing, and provides several directions for future exploration.

Part I

POLYGLOT MODELING

Chapter 2

STRUCTURED PREDICTION

This chapter discusses work originally published in Mulcaire et al. (2018), in collaboration with Swabha Swayamdipta and Noah Smith.

2.1 Introduction

As described in the introduction, different languages have dramatically different amounts of data available. This is true not just for scraped text but also for labeled training data for tasks involving the prediction of linguistic structures; for example, version 2.9 of the Universal Dependencies project has treebanks in 121 languages, but while the German treebanks comprise nearly 3.75 million annotated tokens, more than half of the included languages have fewer than 100,000 tokens, and a third have fewer than 10,000. Furthermore, since these tasks require expert annotators rather than crowd workers, these disparities can be expensive to address.

To address this problem, Ammar et al. (2016a) found that using training data from multiple languages annotated with Universal Dependencies (Nivre et al., 2016), and represented using multilingual word vectors, outperformed monolingual training.

In this chapter, we apply a similar approach to the task of semantic role labeling, and train a monolingual SRL parser, based on that of He et al. (2017), to train it jointly on multiple languages at once. For each language in the CoNLL 2009 SRL dataset (Hajič et al., 2009), we train a parser on that language's data alone, and compare it to a polyglot parser trained on that language and English, by evaluating on the target language test set.

The CoNLL 2009 dataset includes seven different languages, allowing study of trends across the same. The data format is shared; regardless of the language, semantic relations are represented as shallow dependency trees with arcs between predicates and arguments. Unlike the Universal Dependencies dataset used in Ammar et al. (2016a), however, the semantic label spaces are entirely language-specific, making our task more challenging. Nonetheless, the success of polyglot training in this setting demonstrates that sharing of statistical strength across languages does not depend on explicit alignment in annotated data or even identical annotation conventions, and can be done simply through parameter sharing.

We include a breakdown into label categories of the differences between the monolingual and polyglot models. Our findings indicate that polyglot training consistently improves label accuracy for common labels.

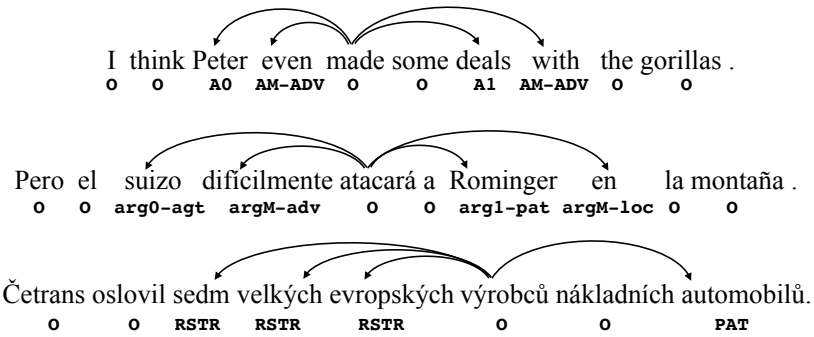

Figure 2.1: Example predicate-argument structures from English, Spanish, and Czech. Note that the argument labels are different in each language.

	# sentences	# sentences w/ 1+ predicates	# predicates
CAT	13200	12876	37444
CES	38727	38579	414133
DEU	36020	14282	17400
ENG	39279	37847	179014
JPN	4393	4344	25712
SPA	14329	13836	43828
ZHO	22277	21073	102827

Table 2.1: Train data statistics. Languages are indicated with ISO 639-3 codes.

2.2 Data

We evaluate our system on the semantic role labeling portion of the CoNLL-2009 shared task (Hajič et al., 2009), on all seven languages, namely Catalan, Chinese, Czech, English, German, Japanese and Spanish. For each language, certain tokens in each sentence in the dataset are marked as predicates. Each predicate takes as arguments other words in the same sentence, their relationship marked by labeled dependency arcs. Sentences may contain zero, one or more predicates.

Despite the consistency of this format, there are significant differences between the training sets across languages, because the datasets were annotated independently under diverse formalisms and only later converted into CoNLL format (Hajič et al., 2009). English uses PropBank role labels (Palmer et al., 2005). Catalan, Chinese, English, German, and Spanish include (but are not limited to) labels such as "arg_0-agt" (for "agent") or "A_0" that may correspond to some degree to each other and to the English roles. Catalan and Spanish share most labels (being drawn from the same source corpus, AnCora; Taulé et al.,

2008), and English and German share some labels. Czech and Japanese each have their own distinct sets of argument labels, most of which do not have clear correspondences to English labels or to each other.

We also note that, due to semi-automatic projection of annotations to construct the German dataset, more than half of German sentences do *not* include labeled predicate and arguments. Thus while the German dataset includes almost as many sentences as Czech, and far more sentences than Japanese, it has by far the fewest actual training examples (predicate-argument structures); see Table 2.1.

2.3 Model

Given a sentence with a marked predicate, the CoNLL 2009 shared task requires disambiguation of the sense of the predicate, and labeling all its dependent arguments. The shared task assumed predicates have already been identified, hence we do not handle the predicate identification task.

Our basic model adapts the span-based dependency SRL model of He et al. (2017). This adaptation treats the dependent arguments as argument spans of length 1. Additionally, BIO consistency constraints are removed from the original model— each token is tagged simply with the argument label or an empty tag. A similar approach has also been proposed by Marcheggiani et al. (2017).

The input to the model consists of a sequence of pretrained embeddings for the surface forms of the sentence tokens. Each token embedding is also concatenated with a vector indicating whether the word is a predicate or not. Since the part-of-speech tags in the CoNLL 2009 dataset are based on a different tagset for each language, we do not use these. Each training instance consists of the annotations for a single predicate. These representations are then passed through a deep, multi-layer bidirectional LSTM (Graves, 2013; Hochreiter and Schmidhuber, 1997) with highway connections (Srivastava et al., 2015).

We use the hidden representations produced by the deep biLSTM for both argument labeling and predicate sense disambiguation in a multitask setup; this is a modification to the models of He et al. (2017), who did not handle predicate senses, and of Marcheggiani et al. (2017), who used a separate model. These two predictions are made independently, with separate softmaxes over different last-layer parameters; we then combine the losses for each task when training. For predicate sense disambiguation, since the predicate has been identified, we choose from a small set of valid predicate senses as the tag for that token. This set of possible senses is selected based on the training data: we map from lemmatized tokens to predicates and from predicates to the set of all senses of that predicate. Most predicates are only observed to have one or two corresponding senses, making the set of available senses at test time quite small (less than five senses/predicate on average across all languages). If a particular lemma was not observed in training, we heuristically predict it as the first sense of that predicate. For Czech and Japanese, the predicate sense

annotation is simply the lemmatized token of the predicate, giving a one-to-one predicate-"sense" mapping.

For argument labeling, every token in the sentence is assigned one of the argument labels, or NULL if the model predicts it is not an argument to the indicated predicate.

2.3.1 *Monolingual Baseline*

We use pretrained word embeddings as input to the model. For each of the shared task languages, we produced GloVe vectors (Pennington et al., 2014) from the news, web, and Wikipedia text of the Leipzig Corpora Collection (Goldhahn et al., 2012).[1] We trained 300-dimensional vectors, then reduced them to 100 dimensions with principal component analysis for efficiency.

Simple Polyglot Sharing In the first polyglot variant, we consider multilingual sharing between each language and English by using pretrained *multilingual* embeddings. This polyglot model is trained on the union of annotations in the two languages. We use stratified sampling to give the two datasets equal effective weight in training, and we ensure that every training instance is seen at least once per epoch.

2.3.2 *Pretrained multilingual embeddings.*

The basis of our polyglot training is the use of pretrained multilingual word vectors, which allow representing entirely distinct vocabularies (such as the tokens of different languages) in a shared representation space, allowing crosslingual learning (Klementiev et al., 2012). We produced multilingual embeddings from the monolingual embeddings using the method of Ammar et al. (2016b): for each non-English language, a small crosslingual dictionary and canonical correlation analysis was used to find a transformation of the non-English vectors into the English vector space (Faruqui and Dyer, 2014).

Unlike multilingual word representations, argument label sets are disjoint between language pairs, and correspondences are not clearly defined. Hence, we use separate label representations for each language's labels. Similarly, while (for example) ENG:look and SPA:mirar may be semantically connected, the senses `look.01` and `mirar.01` may not correspond. For this reason, we also use language-specific predicate sense representations.

2.3.3 *Language Identification*

In the second variant, we concatenate a language ID vector to each multilingual word embedding and predicate indicator feature in the input representation. This vector is ran-

[1]For English we used the vectors provided on the GloVe website `nlp.stanford.edu/projects/glove/`.

domly initialized and updated in training. These additional parameters provide a small degree of language-specificity in the model, while still sharing most parameters.

2.3.4 *Language-Specific LSTMs*

This third variant takes inspiration from the "frustratingly easy" architecture of Daume III (2007) for domain adaptation. In addition to processing every example with a shared biLSTM as in previous models, we add language-specific biLSTMs that are trained only on the examples belonging to one language. Each of these language-specific biLSTMs is two layers deep, and is combined with the shared biSLTM in the input to the third layer. This adds a greater degree of language-specific processing while still sharing representations across languages. It also uses the language identification vector and multilingual word vectors in the input.

2.4 Experiments

We present our results in Table 2.2. We observe that simple polyglot training improves over monolingual training, with the exception of Czech, where we observe no change in performance. The languages with the fewest training examples (German, Japanese, Catalan) show the most improvement, while large-dataset languages such as Czech or Chinese see little or no improvement (Figure 2.2).

The language ID model performs inconsistently; it is better than the simple polyglot model in some cases, including Czech, but not in all. The language-specific LSTMs model performs best on a few languages, such as Catalan and Chinese, but worst on others. While these results may reflect differences between languages in the optimal amount of crosslingual sharing, we focus on the simple polyglot results in our analysis, which sufficiently demonstrate that polyglot training can improve performance over monolingual training.

We also report performance of state-of-the-art systems in each of these languages, all of which make explicit use of syntactic features, Marcheggiani et al. (2017) excepted. While this results in better performance on many languages, our model has the advantage of not relying on a syntactic parser, and is hence more applicable to languages with lower resources. However, the results suggest that syntactic information is critical for strong performance on German, which has the fewest predicates and thus the least semantic annotation for a semantics-only model to learn from. Nevertheless, our baseline is on par with the best published scores for Chinese, and it shows strong performance on most languages.

2.4.1 *Label-wise results.*

Table 2.3 gives the F_1 scores for individual label categories in the Catalan and Spanish datasets, as an illustration of the larger trend. In both languages, we find a small but consistent improvement in the most common label categories (e.g., arg_1 and arg_M). Less com-

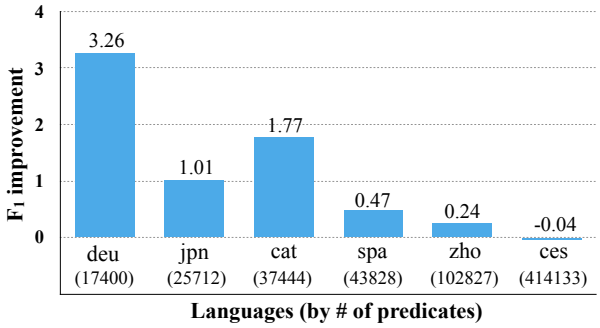

Figure 2.2: Improvement in absolute F_1 with polyglot training with addition of English. Languages are sorted in order of increasing number of predicates in the training set.

Model	CAT	CES	DEU	ENG	JPN	SPA	ZHO
Marcheggiani et al. (2017)	–	86.00	–	87.60	–	80.30	81.20
Best previously reported	*80.32*	86.00	*80.10*	*89.10*	*78.15*	*80.50*	*81.20*
Monolingual	77.31	84.87	66.71	86.54	74.99	75.98	81.26
+ ENG(simple polyglot)	79.08	84.82	69.97	–	76.00	76.45	81.50
+ ENG(language ID)	79.05	85.14	69.49	–	75.77	77.32	81.42
+ ENG(language-specific LSTMs)	79.45	84.78	68.30	–	75.88	76.86	81.89

Table 2.2: Semantic F_1 scores (including predicate sense disambiguation) on the CoNLL 2009 dataset. State of the art for Catalan and Japanese is from Zhao et al. (2009), for German and Spanish from Roth and Lapata (2016), for English and Chinese from Marcheggiani and Titov (2017). Italics indicate use of syntax.

	arg$_0$	**arg**$_1$	**arg**$_2$	**arg**$_3$	**arg**$_4$	**arg**$_L$	**arg**$_M$
Gold label count (CAT)	2117	4296	1713	61	71	49	2968
Monolingual CAT F_1	82.06	79.06	68.95	28.89	42.42	39.51	60.85
+ ENG improvement	+2.75	+2.58	+4.53	+18.17	+9.81	+1.35	+1.10
Gold label count (SPA)	2438	4295	1677	49	82	46	3237
Monolingual SPA F_1	82.44	77.93	70.24	28.89	41.15	22.50	58.89
+ ENG improvement	+0.37	+0.43	+1.35	-3.40	-3.48	+4.01	+1.26

Table 2.3: Per-label breakdown of F_1 scores for Catalan and Spanish. These numbers reflect labels for each argument; the combination is different from the overall semantic F_1, which includes predicate sense disambiguation.

mon label categories are sensitive to small changes in performance; they have the largest

ENG-only	+CAT	+CES	+DEU	+JPN	+SPA	+ZHO
86.54	86.79	87.07	87.07	87.11	87.24	87.10

Table 2.4: Semantic F_1 scores on the English test set for each language pair.

changes in F_1 in absolute value, but without a consistent direction. This could be attributed to the addition of English data, which improves learning of representations that are useful for the most common labels, but is essentially a random perturbation for the rarer ones. This pattern is seen across languages, and consistently results in overall gains from polyglot training.

One exception is in Czech, where polyglot training reduces accuracy on several common argument labels, e.g., PAT and LOC. While the effect sizes are small (consistent with other languages), the overall F_1 score on Czech decreases slightly in the polyglot condition. It may be that the Czech dataset is too large to make use of the comparatively small amount of English data, or that differences in the annotation schemes prevent effective crosslingual transfer.

Future work on language pairs that do not include English could provide further insights. Catalan and Spanish, for example, are closely related and use the same argument label set (both being drawn from the AnCora corpus) which would allow for sharing output representations as well as input tokens and parameters.

Polyglot English results. For each language pair, we also evaluated the simple polyglot model on the English test set from the CoNLL 2009 shared task (Table 2.4). English SRL consistently benefits from polyglot training, with an increase of 0.25–0.7 absolute F_1 points, depending on the language. Surprisingly, Czech provides the smallest improvement, despite the large amount of data added; the absence of crosslingual transfer in both directions for the English-Czech case, breaking the pattern seen in other languages, could therefore be due to differences in annotation rather than questions of dataset size.

Labeled vs. unlabeled F_1. Table 2.5 provides unlabeled F_1 scores for each language pair. As can be seen here, the unlabeled F_1 improvements are generally positive but small, indicating that polyglot training can help both in structure prediction and labeling of arguments. The pattern of seeing the largest improvements on the languages with the smallest datasets generally holds here: the largest F_1 gains are in German and Catalan, followed by Japanese, with minimal or no improvement elsewhere.

Model	CAT	CES	DEU	ENG	JPN	SPA	ZHO
Monolingual	93.92	91.92	87.95	92.87	85.55	93.61	87.93
+ ENG	94.09	91.97	89.01	–	86.17	93.65	87.90

Table 2.5: Unlabeled semantic F_1 scores on the CoNLL 2009 dataset.

2.5 Related Work

The shift to neural architectures has led to significant improvements in SRL, though the pre-neural approach of Zhao et al. (2009), who used syntactic parsers to provide features to their SRL system, remained (in 2017) the best reported result on the Catalan and Japanese portions of CoNLL 2009. Swayamdipta et al. (2016) present a transition-based stack LSTM model that predicts syntax and semantics jointly, as a remedy to the reliance on pipelined models. Guo et al. (2016a) and Roth and Lapata (2016) use deep biLSTM architectures which use syntactic information to guide the composition. Marcheggiani et al. (2017) use a simple LSTM model over word tokens to tag semantic dependencies, like our model. Their model predicts a token's label based on the combination of the token vector and the predicate vector, and saw benefits from using POS tags, both improvements that could be added to our model. Marcheggiani and Titov (2017) apply graph convolutional networks to SRL, obtaining state of the art results on English and Chinese. All of these approaches are orthogonal to ours, and might benefit from polyglot training.

Other multilingual models have been proposed for semantics. Richardson et al. (2018) train on multiple (natural language)-(programming language) pairs to improve a model that translates API text into code signature representations. Duong et al. (2017) treat English and German semantic parsing as a multi-task learning problem and saw improvement over monolingual baselines, especially for small datasets. Most relevant to our work is Johannsen et al. (2015), which trains a polyglot model for *frame*-semantic parsing. In addition to sharing features with multilingual word vectors, they use them to find word translations of target language words for additional lexical features.

2.6 Summary

In this chapter, we described a straightforward method for multilingual training for predicting semantic dependency structures: use multilingual word vectors to represent text from multiple languages in a shared embedding space, and combine supervised training data across languages. This allows sharing without crosslingual alignment of labels, shared annotation, or parallel training data. As our multilingual embeddings were produced by alignment of monolingual embeddings (Faruqui and Dyer, 2014), our method does not even depend on the existence of parallel text corpora.

The ability of the polyglot SRL model to learn shared linguistic information that is applicable to multiple annotation systems is conceptually similar to prior and concurrent work on multitask learning (Swayamdipta et al., 2018; Peng et al., 2018) and to the pretraining-finetuning paradigm, in which an unsupervised objective is used to learn linguistic information that can then be applied to a variety of tasks (Devlin et al., 2019). Future work could explore the ability of polyglot models to automatically identify similarities in existing annotation systems for structured prediction tasks, perhaps informing the development of new annotation schemes.

This chapter is the first application of a polyglot model for this task, and we showed that such a model can outperform a monolingual one for semantic analysis, particularly for languages with less data. In the following chapter, we will see how this approach can be applied to the unsupervised task of language modeling.

Chapter 3

CONTEXTUALIZED REPRESENTATIONS

This chapter discusses work originally published in Mulcaire et al. (2019b) (Sections 3.1 - 3.6) and Mulcaire et al. (2019a) (Sections 3.1, 3.7 and 3.8), in collaboration with Jungo Kasai and Noah Smith.

3.1 Introduction

While the previous chapter described a structured prediction model that made use of word vectors, *contextual* word representations (CWR) extracted from language models (LMs) have advanced the state of the art beyond what was achieved with non-contextual word representations on many monolingual NLP tasks (Peters et al., 2018). In this chapter, we investigate polyglot language modeling and show that contextual word representations can also be made multilingual, to the benefit of downstream tasks.

We introduce a method to produce multilingual CWR by training a single "polyglot" language model on text in multiple languages. As our work is a multilingual extension of ELMo (Peters et al., 2018), we call it Rosita (after a bilingual character from *Sesame Street*). Our hypothesis is that, although each language is unique, different languages manifest similar characteristics (e.g., morphological, lexical, syntactic) which can be exploited by training a single language model with text from multiple languages. Previous work has shown this to be true to some degree in the context of structured prediction tasks such as semantic role labeling (Chapter 2), syntactic dependency parsing (Ammar et al., 2016a), and named entity recognition (Xie et al., 2018), as well as language modeling for phonetic sequences (Tsvetkov et al., 2016) and for speech recognition (Ragni et al., 2016). de Lhoneux et al. (2018), however, found that while parameter sharing between languages can improve performance in dependency parsing, the effect is variable, depending on the language pair and the parameter sharing strategy. Other prior work also reported that in some cases concatenating data from different languages can hurt performance in dependency parsing (Che et al., 2018). These mixed results suggest that while crosslingual transfer in neural network models is a promising direction, the best blend of polyglot and language-specific elements may depend on the task and architecture. We find, however, that contextual representations from polyglot language models succeed in a range of settings, even where multilingual word type embeddings do not, and are a useful technique for crosslingual transfer, producing state-of-the-art results in multiple tasks.

We explore crosslingual transfer between highly dissimilar languages for language modeling and for three downstream tasks: Universal Dependency (UD) parsing, semantic

role labeling (SRL), and named entity recognition (NER). We also experiment with a wider range of languages, including transfer between phylogenetically related languages, on UD parsing only. This is some of the first work using polyglot LMs to produce contextual representations,[1] and the first analysis comparing them to monolingual LMs for this purpose. Our experiments focus on comparisons in three dimensions: monolingual vs. polyglot representations, contextual vs. word type embeddings, and, within the contextual representation paradigm, purely character-based language models vs. ones that include word-level input.

Previous work has shown that contextual representations offer a significant advantage over traditional word embeddings (word type representations). In this chapter, we show that, on these tasks, polyglot character-based language models can provide additional benefits on top of those offered by contextualization. Specifically, even when crosslingual transfer with word type embeddings hurts target language performance relative to monolingual models, polyglot *contextual* representations can improve target language performance relative to monolingual versions, suggesting that polyglot language models tie dissimilar languages in an effective way.

3.2 Polyglot Language Models

We first describe the language models we use for multilingual (and monolingual) CWR.

3.2.1 Pretraining Data and Preprocessing

Because the Universal Dependencies treebanks we use for the parsing task predominantly use Traditional Chinese characters and the Ontonotes data for SRL and NER consist of Simplified Chinese, we train separate language models for the two variants. For English we use text from the Billion Word Benchmark (Chelba et al., 2013), for Traditional Chinese, wiki and web data provided for the CoNLL 2017 Shared Task (Ginter et al., 2017), for Simplified Chinese, newswire text from Xinhua,[2] and for Arabic, newswire text from AFP.[3] We use approximately 60 million tokens of news and web text for each language. We tokenized the language model training data for English and Simplified Chinese using Stanford CoreNLP (Manning et al., 2014). The Traditional Chinese corpus was already presegmented by UDPipe (Ginter et al., 2017; Straka et al., 2016). We found that the Arabic vocabulary from AFP matched both the UD and Ontonotes data reasonably well without additional tokenization. We also processed all corpora to normalize punctuation and remove non-text.

[1] Contemporaneous work used polyglot LMs for natural language inference and machine translation (Lample and Conneau, 2019).

[2] catalog.ldc.upenn.edu/LDC95T13

[3] catalog.ldc.upenn.edu/LDC2001T55

3.2.2 *Training*

We base our language models on the ELMo method (Peters et al., 2018), which encodes each word with a character CNN, then processes the word in context with a word-level LSTM.[4] Following Che et al. (2018), who used 20 million words per language to train monolingual language models for many languages, we use the same hyperparameters used to train the monolingual English language model from Peters et al. (2018), except that we reduce the internal LSTM dimension from 4096 to 2048. For each target language dataset (Traditional Chinese, Simplified Chinese, and Arabic), we produce:

- a monolingual language model with character CNN (MONOCHAR) trained on that language's data;
- a polyglot LM (ROSITACHAR) trained with the same code, on that language's data with an additional, equal amount of English data;
- a modified polyglot LM (ROSITAWORD), described below.

The ROSITAWORD model concatenates a 300 dimensional word type embedding, initialized with multilingual word embeddings, to the character CNN encoding of the word, before passing this combined vector to the bidirectional LSTM. The idea of this word-level initialization is to bias the model toward crosslingual sharing; because words with similar meanings have similar representations, the features that the model learns are expected to be at least partially language-agnostic. The word type embeddings used for these models, as well as elsewhere in this work, are trained on our language model training set using the fastText method (Bojanowski et al., 2017), and target language vectors are aligned with the English ones using supervised MUSE[5] (Conneau et al., 2018). See appendix for more LM training details.

3.3 Experimental setup

All of our task models (UD, SRL, and NER) are implemented in AllenNLP, version 0.7.2 (Gardner et al., 2018).[6] We generally follow the default hyperparameters and training schemes provided in the AllenNLP library regardless of language. See appendix for the

[4] A possible alternative is BERT (Devlin et al., 2019), which uses a bidirectional objective and a transformer architecture in place of the LSTM. Notably, one of the provided BERT models was trained on several languages in combination, in a simple polyglot approach (see https://github.com/google-research/bert/blob/master/multilingual.md). Our initial exploration of multilingual BERT models raised sufficient questions about preprocessing that we defer comparison to future work.

[5] For our English/Chinese and English/Arabic data, their unsupervised method yielded substantially worse results in word translation.

[6] We make our multilingual fork available at https://github.com/pmulcaire/rosita

vectors (lang.)	task lang.	UD LAS	SRL F_1	NER F_1
fastText (CMN)	CMN	$85.15_{\pm 0.12}$	69.79	76.31
fastText (CMN+ENG)	CMN+ENG	$84.92_{\pm 0.28}$	70.82	76.05
MONOCHAR (CMN)	CMN	$87.55_{\pm 0.25}$	74.14	78.18
ROSITACHAR (CMN+ENG)	CMN	$87.16_{\pm 0.08}$	74.24	**78.29**
ROSITACHAR (CMN+ENG)	CMN+ENG	$\mathbf{87.75}_{\pm 0.16}$	74.69	77.68
ROSITAWORD (CMN+ENG)	CMN	$86.50_{\pm 0.17}$	**74.84**	77.19
ROSITAWORD (CMN+ENG)	CMN+ENG	$86.37_{\pm 0.35}$	74.69	77.16
Best prior work	CMN	–	62.83	75.63
fastText (ARA)	ARA	$82.58_{\pm 0.51}$	50.50	71.60
fastText (ARA+ENG)	ARA+ENG	$82.67_{\pm 0.46}$	54.82	71.45
MONOCHAR (ARA)	ARA	$84.98_{\pm 0.18}$	**59.55**	75.02
ROSITACHAR (ARA+ENG)	ARA	$84.98_{\pm 0.12}$	58.69	75.56
ROSITACHAR (ARA+ENG)	ARA+ENG	$\mathbf{85.24}_{\pm 0.13}$	59.29	**76.19**
ROSITAWORD (ARA+ENG)	ARA	$84.34_{\pm 0.20}$	58.34	74.02
ROSITAWORD (ARA+ENG)	ARA+ENG	$84.24_{\pm 0.13}$	59.47	72.79
Best prior work	ARA	–	48.68	68.02

Table 3.1: LAS for UD parsing, F_1 for SRL, and F_1 for NER, with different input representations. For UD, each number is an average over five runs with different initialization, with standard deviation. SRL/NER results are from one run. The "task lang." column indicates whether the UD/SRL/NER model was trained on annotated text in the target language alone, or a blend of English and the target language data. ROSITAWORD LMs use as word-level input the same multilingual word vectors as fastText models. The best prior result for Ontonotes Chinese NER is in Shen et al. (2018); the others are from Pradhan et al. (2013).

complete list of our hyperparameters. For each task, we experiment with five types of word representations: in addition to the three language model types (MONOCHAR, ROSITACHAR, and ROSITAWORD) described above, we show results for the task models trained with monolingual and polyglot non-contextual word embeddings.

After pretraining, the word representations are fine-tuned to the specific task during task training. In non-contextual cases, we fine-tune by updating word embeddings directly, while in contextual cases, we only update coefficients for a linear combination of the internal representation layers for efficiency (Peters et al., 2018). In order to properly evaluate our models' generalization ability, we ensure that sentences in the test data are excluded from the data used to train the language models.

3.4 Universal Dependencies parsing

We use a state-of-the-art graph-based dependency parser with BiLSTM and biaffine attention (Dozat and Manning, 2017). Specifically, the parser takes as input word representations and 100-dimensional fine-grained POS embeddings following Dozat and Manning (2017). We use the same version (2.2) of the Universal Dependencies treebanks and the same train/dev./test splits as the CoNLL 2018 shared task on multilingual dependency parsing (Zeman et al., 2018). Specifically, we use the GUM treebank for English,[7] GSD for Chinese, and PADT for Arabic. For training and validation, we use the provided gold POS tags and word segmentation.

For each configuration, we run experiments five times with random initializations and report the mean and standard deviation. For testing, we use the CoNLL 2018 evaluation script and consider two scenarios: (1) gold POS tags and word segmentations and (2) predicted POS tags and word segmentations from the system outputs of Che et al. (2018) and Qi et al. (2018).[8] The former scenario enables us to purely assess parsing performance; see column 3 in Table 3.1 for these results on Chinese and Arabic. The latter allows for a direct comparison to the best previously reported parsers (Chinese, Che et al., 2018; Arabic, Qi et al., 2018). See Table 3.2 for these results.

As seen in Table 3.1, the Universal Dependencies results generally show a significant improvement from the use of CWR. The best results for both languages come from the ROSITACHAR LM and polyglot task models, showing that polyglot training helps, but that the word-embedding initialization of the ROSITAWORD model does not necessarily lead to a better final model. The results also suggest that combining ROSITACHAR LM and polyglot task training is key to improve parsing performance. Table 3.2 shows that we outperform the state-of-the-art systems from the shared task competition. In particular, our

[7] While there are several UD English corpora, we choose the GUM corpus to minimize domain mismatch.

[8] System outputs for all systems are available at https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2885

LM type	task lang.	LAS
Harbin (Che et al., 2018)	CMN	76.77
Harbin (non-ensemble)	CMN	75.55
ROSITACHAR	CMN	77.40
ROSITACHAR	CMN+ENG	**77.63**
Stanford (Qi et al., 2018)	ARA	77.06
ROSITACHAR	ARA	77.79
ROSITACHAR	ARA+ENG	**78.02**

Table 3.2: LAS (F_1) comparison to the winning systems for each language in the CoNLL 2018 shared task for UD. We use predicted POS and the segmentation of the winning system for that language. The ROSITACHAR LM variant was selected based on development performance in the gold-segmentation condition.

LMs even outperform the Harbin system, which uses monolingual CWR and an ensemble of three biaffine parsers.

3.5 Semantic role labeling

We use a strong existing model based on BIO tagging on top of a deep interleaving BiLSTM with highway connections (He et al., 2017). The SRL model takes as input word representations and 100-dimensional predicate indicator embeddings following He et al. (2017). In this chapter, we use a PropBank-style, span-based SRL dataset for English, Chinese, and Arabic: Ontonotes (Pradhan et al., 2013). Note that unlike the dependency-based CoNLL 2009 dataset used in Chapter 2, Ontonotes provides annotations using a single shared annotation scheme for English, Chinese, and Arabic, which can facilitate crosslingual transfer; this allows us to focus on the effect of the CWRs rather than the annotation scheme. For Chinese and English we simply use the provided surface form of the words. The Arabic text in Ontonotes has diacritics to indicate vocalization which do not appear (or only infrequently) in the original source or in our language modeling data. We remove these for better consistency with the language model vocabulary. We use gold predicates and the CoNLL 2005 evaluation script for the experiments below to ensure our results are comparable to prior work. See column 4 in Table 3.1 for results on the CoNLL-2012 Chinese and Arabic test sets.

The SRL results confirm the advantage of CWR. Unlike the other two tasks, multilingual word type embeddings are better than monolingual versions in SRL. Perhaps relatedly, models using ROSITAWORD are more successful here, providing the highest performance on Chinese. One unusual result is that the model using the MONOCHAR LM is

most successful for Arabic. This may be linked to the poor results on Arabic SRL overall, which are likely due to the much smaller size of the corpus compared to Chinese (less than 20% as many annotated predicates) and higher proportion of language-specific tags. Such language-specific tags in Arabic could limit the effectiveness of shared English-Arabic representations. Still, polyglot methods' performance is only slightly behind.

3.6 Named entity recognition

We use the state-of-the-art BiLSTM-CRF NER model with the BIO tagging scheme (Peters et al., 2017). The network takes as input word representations and 128-dimensional character-level embeddings from a character LSTM. We again use the Ontonotes dataset with the standard data splits. See the last column in Table 3.1 for results on the CoNLL-2012 Chinese and Arabic test sets. As with most other experiments, the NER results show a strong advantage from the use of contextual representations and a smaller additional advantage from those produced by polyglot LMs.

3.7 Comparing joint crosslingual training to post-hoc alignment

Other work has extended contextual word representations (CWRs) multilingually by aligning multiple monolingual language models crosslingually (*retrofitting* approach; Schuster et al., 2019; Aldarmaki and Diab, 2019). In this section, we compare this method to the joint crosslingual training method described in Section 3.2 using the same LM training data, and discover that the joint training approach generally yields better performance on the downstream task of low-resource dependency parsing, even without crosslingual supervision. We also apply multilingual CWRs produced by the joint training approach to a more diverse set of languages, and show that while it is still effective in transfer between distant languages, phylogenetically related source languages are generally more helpful.

Retrofitted contextualized word representations Following Schuster et al. (2019), we first train a bidirectional LM with two-layer LSTMs on top of character CNNs for each language (ELMo, Peters et al., 2018), and then align the monolingual LMs across languages. Denote the hidden state in the jth layer for word i in context c by $\mathbf{h}_{i,c}^{(j)}$. We use a trainable weighted average of the three layers (character-CNN and two LSTM layers) to compute the contextual representation $\mathbf{e}_{i,c}$ for the word: $\mathbf{e}_{i,c} = \sum_{j=0}^{2} \lambda_j \mathbf{h}_{i,c}^{(j)}$ (Peters et al., 2018).[9] In the first step, we compute an "anchor" $\mathbf{h}_i^{(j)}$ for each word by averaging $\mathbf{h}_{i,c}^{(j)}$ over all occurrences in an LM corpus. We then apply a standard dictionary-based technique[10] to create

[9] Schuster et al. (2019) only used the first LSTM layer, but we found a performance benefit from using all layers in preliminary results.

[10] Conneau et al. (2018) developed an unsupervised alignment technique that does not require a dictionary. We found that their unsupervised alignment yielded substantially degraded performance in downstream

multilingual word embeddings (Mikolov et al., 2013; Conneau et al., 2018). In particular, suppose that we have a word-translation dictionary from source language s to target language t. Let $\mathbf{H}_s^{(j)}, \mathbf{H}_t^{(j)}$ be matrices whose columns are the anchors in the jth layer for the source and corresponding target words in the dictionary. For each layer j, find the linear transformation $\mathbf{W}^{*(j)}$ such that

$$\mathbf{W}^{*(j)} = \arg\min_{\mathbf{W}} ||\mathbf{W}\mathbf{H}_s^{(j)} - \mathbf{H}_t^{(j)}||_F$$

The linear transformations are then used to map the LM hidden states for the source language to the target LM space. Specifically, contextual representations for the source and target languages are computed by $\sum_{j=0}^{2} \lambda_j \mathbf{W}^{*(j)} \mathbf{h}_{i,c}^{(j)}$ and $\sum_{j=0}^{2} \lambda_j \mathbf{h}_{i,c}^{(j)}$ respectively. We use publicly available dictionaries from Conneau et al. (2018)[11] and align all languages to the English LM space, again following Schuster et al. (2019).

Refinement after Joint Training It is possible to combine retrofitting with joint crosslingual training; the alignment procedure used in the retrofitting approach can serve as a refinement step on top of an already-polyglot language model. We will see only a limited gain in parsing performance from this refinement in our experiments, suggesting that polyglot LMs are already producing high-quality multilingual CWRs even without crosslingual dictionary supervision.

FastText Baseline We also compare the multilingual CWRs to a subword-based, non-contextual word embedding baseline. We train 300-dimensional word vectors on the same LM data using the fastText method (Bojanowski et al., 2017), and use the same bilingual dictionaries to align them (Conneau et al., 2018).

3.7.1 Dependency Parsers

As in Section 3.4, we train polyglot parsers for multiple languages on top of multilingual CWRs. All parser parameters are shared between the source and target languages. Prior work (Ammar et al., 2016a) suggest that sharing parameters between languages can alleviate the low-resource problem in syntactic parsing, but their experiments are limited to (relatively similar) European languages. Here we explore a wider range of languages, and analyze the particular efficacy of a crosslingual approach to dependency parsing in a low-resource setting.

We use the same dependency parser as in Section 3.4 (Dozat and Manning, 2017), which is also used in Schuster et al. (2019). Note that in these experiments, we use only word representations as input. Universal parts of speech have been shown useful for low-resource

parsing in line with the findings of Schuster et al. (2019).

[11] https://github.com/facebookresearch/MUSE#ground-truth-bilingual-dictionaries

dependency parsing (Duong et al., 2015; Ammar et al., 2016a; Ahmad et al., 2019), but many realistic low-resource scenarios lack reliable part-of-speech taggers; here, we do not use parts of speech as input, and thus avoid the error-prone part-of-speech tagging pipeline. For the fastText baseline, word embeddings are not updated during training, to preserve crosslingual alignment (Ammar et al., 2016a).

3.7.2 *Zero-Target Dependency Parsing*

Following prior work on low-resource dependency parsing and crosslingual transfer (Zhang and Barzilay, 2015; Guo et al., 2015; Ammar et al., 2016a; Schuster et al., 2019), we conduct multi-source experiments on six languages (German, Spanish, French, Italian, Portuguese, and Swedish) from Google universal dependency treebank version 2.0 (McDonald et al., 2013).[12] We train language models on the six languages and English to produce multilingual CWRs. For each tested language, we train a polyglot parser with the multilingual CWRs on the five other languages and English, and apply the parser to the test data for the target language. Importantly, the parsing annotation scheme is shared among the seven languages. Our results will show that the joint training approach for CWRs substantially outperforms the retrofitting approach.

3.7.3 *Diverse Low-Resource Parsing*

The previous experiment compares the joint training and retrofitting approaches in low-resource dependency parsing only for relatively similar languages. In order to study the effectiveness more extensively, we apply it to a more typologically diverse set of languages. We use five pairs of languages for "low-resource simulations," in which we reduce the size of a large treebank, and four languages for "true low-resource experiments," where only small UD treebanks are available, allowing us to compare to other work in the low-resource condition (Table 3.3). Following de Lhoneux et al. (2018), we selected these language pairs to represent linguistic diversity. For each target language, we produce multilingual CWRs by training a polyglot language model with its related language (e.g., Arabic and Hebrew) as well as English (e.g., Arabic and English). We then train a polyglot dependency parser on each language pair and assess the crosslingual transfer in terms of target parsing accuracy.

Each pair of related languages shares features like word order, morphology, or script. For example, Arabic and Hebrew are similar in their rich transfixing morphology (de Lhoneux et al., 2018), and Dutch and German share most of their word order features. We chose Chinese and Japanese as an example of a language pair which does *not* share a language family but does share characters.

---

[12] http://github.com/ryanmcd/uni-dep-tb

Lang	Code	Genus	WALS 81A	Size
English	ENG	Germanic	SVO	–
Arabic	ARA	Semitic	VSO/SVO	5241
Hebrew	HEB	Semitic	SVO	
Croatian	HRV	Slavic	SVO	6983
Russian	RUS	Slavic	SVO	
Dutch	NLD	Germanic	SOV/SVO	12269
German	DEU	Germanic	SOV/SVO	
Spanish	SPA	Romance	SVO	12543
Italian	ITA	Romance	SVO	
Chinese	CMN	Chinese	SVO	3997
Japanese	JPN	Japanese	SOV	
Hungarian	HUN	Ugric	SOV/SVO	910
Finnish	FIN	Finnic	SVO	12217
Vietnamese	VIE	Viet-Muong	SVO	1400
Uyghur	UIG	Turkic	SOV	1656
Kazakh	KAZ	Turkic	SOV	31
Turkish	TUR	Turkic	SOV	3685

Table 3.3: List of the languages used in our UD v2.2 experiments. Each shaded/unshaded section corresponds to a pair of "related" languages. WALS 81A denotes Feature 81A in WALS, Order of Subject, Object, and Verb (Dryer and Haspelmath, 2013) except in the case of Kazakh, for which this feature is not present in WALS; we list it as SOV based on Muhamedowa (2015). "Size" represents the downsampled size in # of sentences used for source treebanks. The four languages in bold face are truly low resource languages (< 2000 trees).

We chose Hungarian, Vietnamese, Uyghur, and Kazakh as true low-resource target languages because they had comparatively small amounts of annotated text in the UD corpus (Vietnamese: 1,400 sentences, 20,285 tokens; Hungarian: 910 sentences, 20,166 tokens; Uyghur: 1,656 sentences, 19,262 tokens; Kazakh: 31 sentences, 529 tokens;), yet had convenient sources of text for LM pretraining (Zeman et al., 2018).[13] Other small treebanks exist, but in most cases another larger treebank exists for the same language, making domain adaptation a more likely option than crosslingual transfer. Also, recent work (Che et al., 2018) using contextual embeddings was top-ranked for most of these languages in the CoNLL 2018 shared task on UD parsing (Zeman et al., 2018).[14] Future work could explore an even more low-resource scenario in which both syntactic annotation and high-quality

[13]The one exception is Uyghur where we only have 3M words in the raw LM data from Zeman et al. (2018).

[14]In Kazakh, Che et al. (2018) did not use CWRs due to the extremely small treebank size.

Model	DEU	SPA	FRA	ITA	POR	SWE	AVG
Schuster et al. (2019) (retrofit)	61.4	77.5	**77.0**	77.6	73.9	71.0	73.1
Schuster et al. (2019) (retrofit -dict.)	**61.7**	76.6	76.3	**77.1**	69.1	54.2	69.2
fastText + Alignment	45.2	68.5	62.8	58.9	61.1	50.4	57.8
ELMos + Alignment (retrofit)	57.3	75.4	73.7	71.6	75.1	74.2	71.2
Rosita (joint train, no dict.)	58.0	**81.8**	75.6	74.8	**77.1**	76.2	73.9
Rosita + Refinement (joint train+retrofit)	**61.7**	79.7	75.8	76.0	76.8	**76.7**	**74.5**

Table 3.4: Zero-target results in LAS. Results reported in prior work (above the line) use an unknown amount of LM training data; all models below the line are limited to approximately 50M words per language.

text for LM pretraining are limited.

For these experiments, we again use the treebank version and splits from the CoNLL 2018 shared task (Zeman et al., 2018).[15] The UD annotation scheme is shared across languages, which facilitates crosslingual transfer. For each triple of two related languages and English, we downsample training and development data to match the language with the smallest treebank size. This allows for fairer comparisons because within each triple, the source language for any parser will have the same amount of training data. We further downsample sentences from the target train/development data to simulate low-resource scenarios. The ratio of training and development data is kept 5:1 throughout the simulations, and we denote the number of sentences in training data by $|D_\tau|$. For testing, we use the CoNLL 2018 script on the gold word segmentations. For the truly low-resource languages, we also present results with word segmentations from the system outputs of Che et al. (2018) (HUN, VIE, UIG) and Smith et al. (2018) (KAZ) for a direct comparison to those languages' best previously reported parsers.[16]

3.8 Results and discussion

In this section we describe the results of the various parsing experiments from the previous section.

3.8.1 Zero-Target Parsing

Table 3.4 shows results on zero-target dependency parsing. First, we see that all CWRs greatly improve upon the fastText baseline. The joint training approach (Rosita), which

[15] See Appendix A for a list of UD treebanks used for each language.

[16] System outputs for all shared task systems are available at https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2885

Model	DEU	SPA	FRA	ITA	POR	SWE	AVG
Zhang and Barzilay (2015)	54.1	68.3	68.8	69.4	72.5	62.5	65.9
Guo et al. (2016b)	55.9	73.1	71.0	71.2	78.6	69.5	69.9
Ammar et al. (2016a)	57.1	74.6	73.9	72.5	77.0	68.1	70.5
Schuster et al. (2019) (retrofit)	**65.2**	80.0	**80.8**	**79.8**	82.7	75.4	77.3
Schuster et al. (2019) (retrofit -dict.)	64.1	77.8	79.8	79.7	79.1	69.6	75.0
Rosita (joint train, no dict.)	63.6	**83.4**	78.9	77.8	83.0	**79.6**	77.7
Rosita + Refinement (joint train+retrofit)	64.8	82.1	78.7	78.8	**84.1**	79.1	**77.9**

Table 3.5: Zero-target results in LAS with gold UPOS.

uses no dictionaries, consistently outperforms the dictionary-dependent retrofitting approach (ELMos+Alignment). As discussed in the previous section, we can apply the alignment method to refine the already-polyglot Rosita using dictionaries. However, we observe a relatively limited gain in overall performance (74.5 vs. 73.9 LAS points), suggesting that Rosita (polyglot language model) is already developing useful multilingual CWRs for parsing without crosslingual supervision. Note that the degraded overall performance of our ELMo+Alignment compared to Schuster et al. (2019)'s reported results (71.2 vs. 73.1) is likely due to the significantly reduced amount of LM data we used in all of our experiments (50M words per language, an order of magnitude reduction from the full Wikipedia dumps used in Schuster et al. (2019)). Schuster et al. (2019) (no dictionaries) is the same retrofitting approach as ELMos+Alignment except that the transformation matrices are learned in an unsupervised fashion without dictionaries (Conneau et al., 2018). The absence of a dictionary yields much worse performance (69.2 vs. 73.1) in contrast with the joint training approach of Rosita, which also does not use a dictionary (73.9).

We also present results using gold universal part of speech to compare to previous work in Table 3.5. We again see Rosita's effectiveness and a marginal benefit from refinement with dictionaries. It should also be noted that the reported results for French, Italian and German in Schuster et al. (2019) outperform all results from our controlled comparison; this may be due to the use of abundant LM training data. Nevertheless, joint training, with or without refinement, performs best on average in both gold and predicted POS settings.

3.8.2 Diverse Low-Resource Parsing

Low-Resource Simulations Figure 3.1 shows simulated low-resource results.[17] Of greatest interest are the significant improvements over monolingual parsers when adding English or related-language data. This improvement is consistent across languages and suggests that crosslingual transfer is a viable solution for a wide range of languages, even

[17] A table with full details including different size simulations is provided in the appendix.

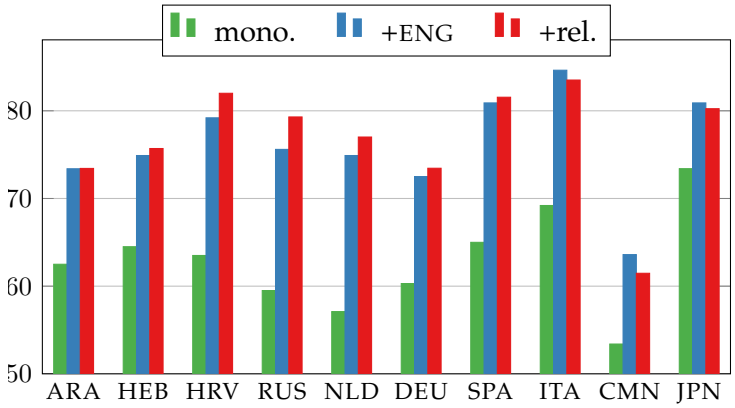

Figure 3.1: LAS for UD parsing results in a simulated low-resource setting where the size of the target language treebank ($|D_\tau|$) is set to 100 sentences.

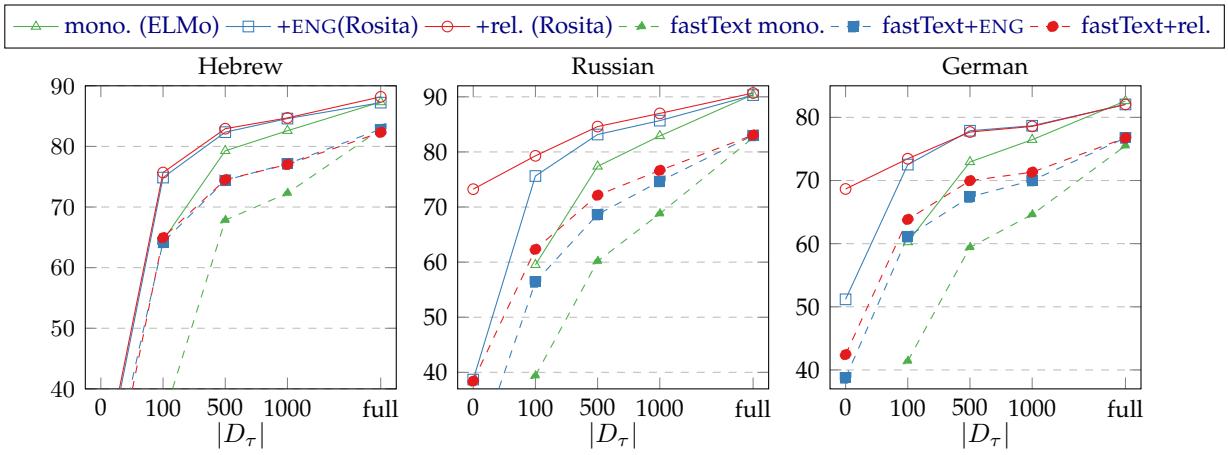

Figure 3.2: Plots of parsing performance vs. target language treebank size for several example languages. The size 0 target treebank point indicates a parser trained *only* on the source language treebank but with polyglot representations, allowing transfer to the target test treebank using no target language training trees. See Appendix A for results with zero-target-treebank and intermediate size data ($|D_\tau| \in \{0, 100, 500, 1000\}$) for all languages.

when (as in our case) language-specific tuning or annotated resources like parallel corpora or bilingual dictionaries are not available. See Figure 3.2 for a visualization of the differences in performance with varying training size. The polyglot advantage is minor when the target language treebank is large, but dramatic in the condition where the target language has only 100 sentences. The fastText approaches consistently underperform the language model approaches, but show the same pattern.

In addition, related-language polyglot ("+rel.") outperforms English polyglot in most cases in the low-resource condition. The exceptions to this pattern are Italian (whose treebank is of a different genre from the Spanish one), and Japanese and Chinese, which differ significantly in morphology and word order. The CMN/JPN result suggests that such typological features influence the degree of crosslingual transfer more than orthographic properties like shared characters. This result in crosslingual transfer also mirrors the observation from prior work (Gerz et al., 2018b) that typological features of the language are predictive of *monolingual* LM performance. The related-language improvement also vanishes in the full-data condition (Figure 3.2), implying that the importance of shared linguistic features can be overcome with sufficient annotated data. It is also noteworthy that variations in word order, such as the order of adjective and noun, do not affect performance: Italian, Arabic, and others use a noun-adjective order while English uses an adjective-noun order, but their +ENG and +rel. results are comparable.

The Croatian and Russian results are notable because of their close phylogenetic relation but different scripts. Though Croatian uses the Latin alphabet and Russian uses Cyrillic, transfer between HRV+RUS is clearly more effective than HRV+ENG (82.00 vs. 79.21 LAS points when $|D_\tau| = 100$). This suggests that character-based LMs can implicitly learn to transliterate between related languages with different scripts, even without parallel supervision.

Truly Low Resource Languages Finally we present "true low-resource" experiments for four languages in which little UD data is available (see Section 3.7.3). Table 3.6 shows these results. Consistent with our simulations, our parsers on top of Rosita (multilingual CWRs from the joint training approach) substantially outperform the parsers with ELMos (monolingual CWRs) in all languages, and establish a new state of the art in Hungarian, Vietnamese, and Kazakh. Consistent with our simulations, we see that training parsers with the target's related language is more effective than with the more distant language, English. It is particularly noteworthy that the Rosita models, which do not use a parallel corpus or dictionary, dramatically improve over the best previously reported result from Schuster et al. (2019) when either the related language of Turkish (51.96 vs. 36.98) or even the more distant language of English (46.03 v.s. 36.98) is used. Schuster et al. (2019) aligned the monolingual ELMos for Kazakh and Turkish using the KAZ-TURdictionary that Rosa and Mareček (2018) derived from parallel text. This result further corroborates our finding that the joint training approach to multilingual CWRs is more effective than retrofitting

Model	gold	pred.
Hungarian (HUN)		
Che et al. (2018) (HUN, ensemble)	–	82.66
Che et al. (2018) (HUN)	–	80.96
ELMo (HUN)	81.89	81.54
Rosita (HUN+ENG)	85.34	84.89
Rosita (HUN+FIN)	**85.40**	**84.96**
Vietnamese (VIE)		
Che et al. (2018) (VIE, ensemble)	–	55.22
ELMo (VIE)	62.67	55.72
Rosita (VIE+ENG)	**63.07**	**56.42**
Uyghur (UIG)		
Che et al. (2018) (UIG, ensemble)	–	**67.05**
Che et al. (2018) (UIG)	–	66.20
ELMo (UIG)	66.64	63.98
Rosita (UIG+ENG)	67.85	65.55
Rosita (UIG+TUR)	**68.08**	65.73
Rosa and Mareček (2018) (KAZ+TUR)	–	26.31
Smith et al. (2018) (KAZ+TUR)	–	31.93
Schuster et al. (2019) (KAZ+TUR)	–	36.98
Rosita (KAZ+ENG)	48.02	46.03
Rosita (KAZ+TUR)	**53.98**	**51.96**

Table 3.6: LAS (F_1) comparison for truly low-resource languages. The gold and pred. columns show results under gold segmentation and predicted segmentation. The languages in the parentheses indicate the languages used in parser training.

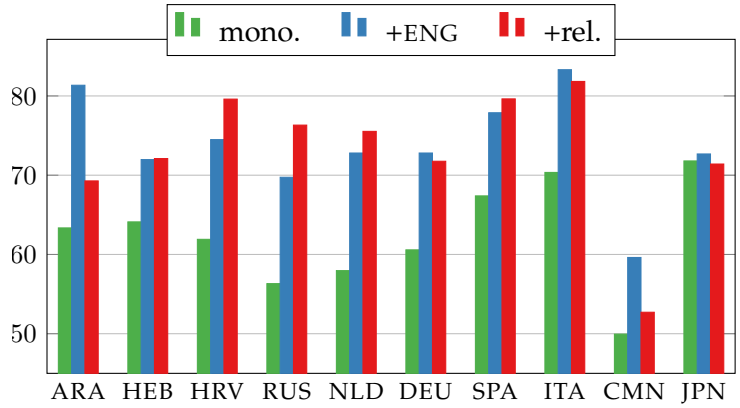

Figure 3.3: LAS for UD parsing results in a simulated low-resource setting ($(|D_\tau| = 100)$ using multilingual BERT embeddings in place of Rosita. Cf. Figure 3.1.

monolingual LMs.

3.8.3 Comparison to Multilingual BERT Embeddings

We also evaluate the diverse low-resource language pairs using pretrained multilingual BERT (Devlin et al., 2019) as text embeddings (Figure 3.3). Here, the same language model (multilingual cased BERT,[18] covering 104 languages) is used for all parsers, with the only variation being in the training treebanks provided to each parser. Parsers are trained using the same hyperparameters and data as in Section 3.7.3.[19]

There are two critical differences from our previous experiments: multilingual BERT is trained on much larger amounts of Wikipedia data compared to other LMs used in this work, and the WordPiece vocabulary (Wu et al., 2016) used in the cased multilingual BERT model has been shown to have a distribution skewed toward Latin alphabets (Ács, 2019). These results are thus not directly comparable to those in Figure 3.1; nevertheless, it is interesting to see that the results obtained with ELMo-like LMs are comparable to and in some cases better than results using a BERT model trained on over a hundred languages. Our results broadly fit with those of Pires et al. (2019), who found that multilingual BERT was useful for zero-shot crosslingual syntactic transfer. In particular, we find nearly no performance benefit from cross-script transfer using BERT in a language pair (English-Japanese) for which they reported poor performance in zero-shot transfer, contrary to our

[18] Available at https://github.com/google-research/bert/

[19] AllenNLP version 0.9.0 was used for these experiments.

results using Rosita (Section 3.8.2).

3.9 Further related work

In addition to the work mentioned above, much previous work has proposed techniques to transfer knowledge from a high-resource to a low-resource language for dependency parsing. Many of these methods use an essentially (either lexicalized or delexicalized) joint polyglot training setup (e.g., McDonald et al., 2011; Cohen et al., 2011; Duong et al., 2015; Guo et al., 2016b; Vilares et al., 2016; Falenska and Çetinoğlu, 2017 as well as many of the CoNLL 2017/2018 shared task participants: Lim and Poibeau (2017); Vania et al. (2017); de Lhoneux et al. (2017); Che et al. (2018); Wan et al. (2018); Smith et al. (2018); Lim et al. (2018)). Some use typological information to facilitate crosslingual transfer (e.g., Naseem et al., 2012; Täckström et al., 2013; Zhang and Barzilay, 2015; Wang and Eisner, 2016; Rasooli and Collins, 2017; Ammar et al., 2016a). Others use bitext (Zeman et al., 2018), manually-specified rules (Naseem et al., 2012), or surface statistics from gold universal part of speech (Wang and Eisner, 2018a,b) to map the source to target. One advantage of the joint training method for producing multilingual CWRs examined in this chapter is that does not rely on such external information about the languages, and instead uses relatively abundant LM data to learn crosslinguality that abstracts away from typological divergence.

3.10 Summary

Overall, our results show that polyglot language models produce representations that are useful for downstream NLP tasks.

While our structured prediction results in Sections 3.4, 3.5, and 3.6 show models using contextual representations consistently outperform those using word type representations, the advantage from polyglot training in some cases is minor, while others (Chinese SRL and Arabic NER) show strong improvement both from contextual word representations and from polyglot training. Thus, while the benefit of crosslingual transfer appears to be somewhat variable and task dependent, it is clear that polyglot training is helpful overall for contextual word representations. Notably, the ROSITACHAR LM does not involve any direct supervision of tying two languages together, such as bilingual dictionaries or parallel corpora, yet is still most often able to learn the most effective representations. One explanation is that it automatically learns crosslingual connections from unlabeled data alone. Another possibility, though, which these experiments cannot rule out, is that the additional data provided in polyglot training simply produces a useful regularization effect by adding "noise" to the input, improving the target language representations without real crosslingual sharing beyond that induced by shared vocabulary, e.g., borrowings, numbers, or punctuation.

Our parsing results in Section 3.8 illustrate that a joint training approach for polyglot language models outperforms a retrofitting approach of aligning monolingual language models. We also see that transfer from related languages may often be preferable to transfer from a more distant language such as English.

These results provide a strong basis for multilingual representation learning and for further study of crosslingual transfer in a low-resource setting beyond dependency parsing.

Part II

VOCABULARY INDEPENDENCE

Chapter 4

GROUNDED COMPOSITIONAL OUTPUT EMBEDDINGS

This chapter discusses work originally published in Pappas et al. (2020), in collaboration with Nikolaos Pappas and Noah Smith.

4.1 Introduction

Language models (LMs) are at the heart of natural language processing, especially following their success in the pretraining paradigm (Dai and Le, 2015; Peters et al., 2018; Devlin et al., 2019; Radford et al., 2019, *inter alia*). Continued advances in NLP rely on the adaptability of LMs to domains beyond their training data and to new domains and tasks, e.g., through domain adaptive pretraining followed by finetuning (Gururangan et al., 2020). Here, we focus on an important component of LMs, namely the **output vocabulary**—over which a LM's probability distribution for the "next word," given the history, ranges—and investigate the impact of the type of its representation on the adaptability of neural LMs.

Today, LMs are typically trained with a closed output vocabulary derived from the training data; the vocabulary is not modified when the language model is adapted or deployed. This makes large pretrained language models struggle with rare words, despite being able to produce contextualized representations for them (Schick and Schütze, 2020). More importantly, this means a generative LM can never give nonzero probability to a specific word it did not see in training. This is a longstanding challenge of language modeling (Jelinek, 1997), but it becomes especially important when we adapt to new domains, tasks, or languages.

To address this, we propose a new word-level language model using Grounded Compositional outputs (GroC) which applies a compositional representation to the output vocabulary (Section 4.4). Each word's output embedding is built from its surface character sequence and (if available) those of semantically related words and a free-text definition of from WordNet (Fellbaum, 1998). We evaluate GroC on language modeling with both fixed and open vocabularies in English, and observe that our model has superior perplexity and is more sample efficient than a variety of existing output embedding approaches, including the adaptive embedding of Baevski and Auli (2019). In cross-domain settings it also outperforms strong interpolated baselines, including the unbounded neural cache model of Grave et al. (2017a) on "near" domains and performs competitively on "far" domains.

4.2 Preliminaries on Language Modeling

Language models assign probability to sequences of tokens; the task is usually framed as learning the conditional probability distributions over individual tokens given their histories of tokens to the left (Bahl et al., 1983). Training requires a sequence of T tokens $\boldsymbol{x} = \langle x_1, \ldots, x_T \rangle$, each x_t a member of a preselected vocabulary $\mathcal{V}$. We let $\mathbf{x}_t \in \{0,1\}^{|\mathcal{V}|}$ denote the one-hot encoding of x_t. The probability of the sequence $\boldsymbol{x}$ is factored using the chain rule of probability:

$$p(\boldsymbol{x}) = \prod_{t=1}^{T} p(x_t \mid x_1, \ldots, x_{t-1}). \tag{4.1}$$

To approximate this joint distribution, researchers have fit parametric families based on relative frequencies (Bahl et al., 1983; Kneser and Ney, 1995; Goodman, 2001) and neural networks (Bengio et al., 2003; Mikolov et al., 2010). Here, we focus on the latter due to their established effectiveness (Merity et al., 2018; Baevski and Auli, 2019). Tokens in this work correspond to words but they can also correspond to individual characters (Al-Rfou et al., 2019) or byte pairs (Radford et al., 2019).

4.2.1 Neural Language Models

To make clear this part's contributions, we describe neural language models by decomposing them into several abstract parts.

In most neural language models, the first layer of computation obtains an input embedding of each history word x_j using a lookup function. In our notation, this corresponds to selecting the word type's row in a fixed **input embedding matrix**, $\mathbf{E}^{in}$: $\mathbf{x}_j^\top \mathbf{E}^{in}$, which we denote $\mathbf{e}^{in}_{x_j}$. Importantly, however, input embeddings need not be lookups; for example, they can be built compositionally from the characters in the surface form of the word (Ling et al., 2015), an idea central to this work.

Next, the history or "prefix" words $\boldsymbol{x}_{<t} = \langle x_1, \ldots, x_{t-1} \rangle$ is encoded into a fixed, d-dimensional vector $\mathbf{h}_{t-1}$ using a **prefix function** $f : \mathcal{V}^* \to \mathbb{R}^d$. f can be a recurrent or feed-forward network; we will experiment with LSTMs (Hochreiter and Schmidhuber, 1997) in Section 4.5, but our method is agnostic to the prefix function design. In general, each history encoding is defined:

$$\mathbf{h}_{t-1} = f(\mathbf{e}^{in}_{x_1}, \ldots, \mathbf{e}^{in}_{x_{t-1}}). \tag{4.2}$$

Finally, the distribution over the next word (random variable X_t) is given by

$$p(X_t = x_t \mid \mathbf{h}_{t-1}) \propto \exp\left(\mathbf{E}^{out}\mathbf{h}_{t-1} + \mathbf{b}\right), \tag{4.3}$$

where $\mathbf{E}^{out} \in \mathbb{R}^{|\mathcal{V}| \times d}$ is the **output embedding matrix** and $\mathbf{b} \in \mathbb{R}^{|\mathcal{V}|}$ is the bias vector (corresponding roughly to unigram log-frequencies of words in the vocabulary).

The parameters of the model—including all parameters of the prefix function f, $\mathbf{E}^{in}$, $\mathbf{E}^{out}$, and $\mathbf{b}$—are all chosen by maximizing the likelihood of the training sequence $\boldsymbol{x}$. Note that, though we focus on an autoregressive (left-to-right) language model objective, our analysis below is applicable to other language model pretraining objectives such as masked language modeling (Devlin et al., 2019) and replaced token detection (Clark et al., 2020).

4.3 Choice of Output Representations

Above we assumed an output embedding matrix $\mathbf{E}^{out}$ that independently parameterizes each word in the vocabulary with a separate d-dimensional vector. This approach requires $d \times |\mathcal{V}|$ parameters, leading to concerns about cost and overparameterization. Prior work addressed this issue by tying parameters between the input and output embedding matrices (i.e., $\mathbf{E}^{out} = \mathbf{E}^{in}$; Inan et al., 2017; Press and Wolf, 2017). However, the parameters for each word are still independent from each other, as displayed in Figure 4.1(a). Updates for the input and output vector are combined, but still occur only in proportion to the frequency of that word.

An alternative, also considered here, is to share output parameters across words as well as with the input embeddings. Specifically, this involves making the output embedding a function of the input embedding using a shared parameterization across words, $\mathbf{E}^{out} = g(\mathbf{E}^{in})$, as displayed in Figure 4.1(b). For example, Gulordava et al. (2018) used a linear transformation, while Baevski and Auli (2019) used a linear transformation for each frequency bin to dedicate parameters to words proportional to their frequencies. Pappas and Henderson (2019) used a deep residual transformation as g, demonstrating that shared parameterizations perform better than independent ones. The two latter studies also provided evidence that models with shared parameterization are more *sample efficient* than independent parameterizations since they perform better on low-frequency words.

Limitations We argue that dependence of a model's parameterization on the size of the vocabulary leads to several limitations shared by current word-level language models. First, the output embedding methods above have terms that scale with the vocabulary size, such as the lookup table for the input embedding or the bias vector, which is a concern for the parameterization of infrequent words. Second, handling of words *unseen* in the training data leads us to the convention of uninformative "out-of-vocabulary" word types or linguistically naïve, data-driven vocabulary transformations that aggressively decompose words into smaller units (Sennrich et al., 2016).

Finally, when pretrained language models are adapted on a downstream task, they do not allow graceful modifications to the vocabulary as required by the task or its data domain. Decoupling the training vocabulary from the target vocabulary that a model uses during inference or finetuning will simplify sequential training and enable open vocabularies.

Building on encouraging results with compositional *input* embeddings (Ling et al.,

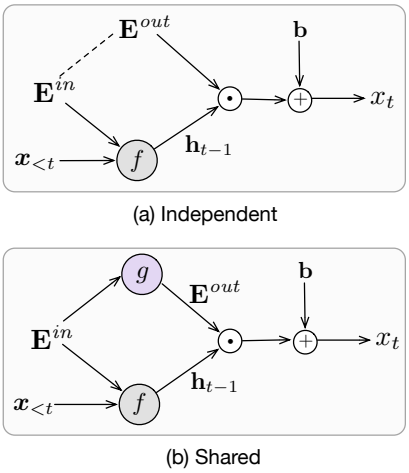

Figure 4.1: Existing output layer parameterizations using independent or shared parameters in the output embedding $\mathbf{E}^{out}$ across words drawn from a vocabulary selected a priori.

2015; Jozefowicz et al., 2016; Peters et al., 2018), we introduce a language model with shared compositional embeddings for input as well as for *output* word representations. Further, we go beyond past work based on surface forms, making optional use of relations and natural language definitions from structured lexicons like WordNet (Fellbaum, 1998). To our knowledge, this is the first word-level language model whose parameters do not depend on the vocabulary size and which is grounded to an external structured lexicon. Our experiments show that our models are more sample efficient (Section 4.5) on closed vocabularies and perform competitively on cross-domain settings (Section 4.6).

4.4 GroC: Grounded Compositional Output Language Models

We present our grounded compositional output language model (Figure 4.2).[1] Following the decomposition of neural language models in Section 4.2 (Equations 4.2–4.3), we consider each part of the model in turn: input embeddings (Section 4.4.1), output embeddings (Section 4.4.2), and bias (Section 4.4.2). As noted above, our approach is agnostic to the training vocabulary ($\mathcal{V}$) and to the prefix encoder (f) that has been the focus of most innovations in neural language model design.

4.4.1 Compositional Input Embeddings

We build on the compositional model of Ling et al. (2015), which encodes a word using its surface string (i.e., character sequence), adding two more sources of information. Peters et al. (2019) enhanced word representations with information from external relational knowledge bases, specifically for words that refer to entitites. Like them, we use a struc-

[1]Code: https://github.com/Noahs-ARK/groc

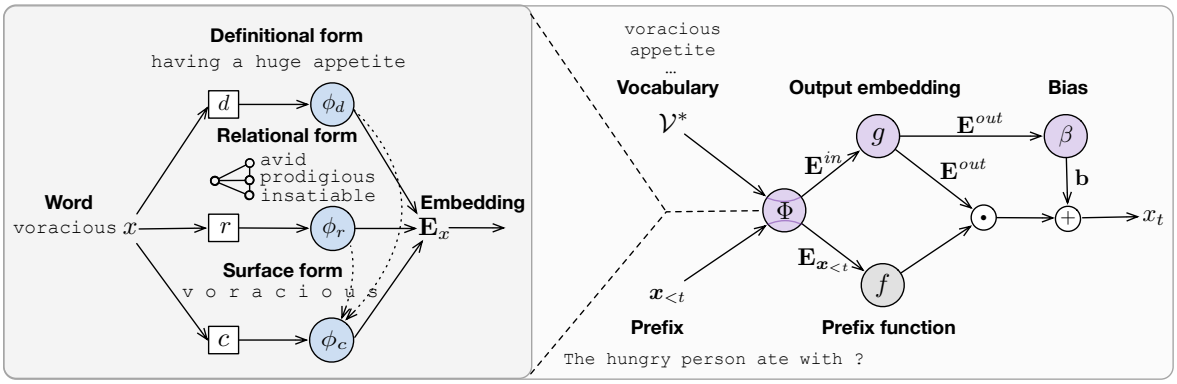

Figure 4.2: Grounded compositional output language modeling. (*Left*) The compositional input embedding is grounded in surface, relational, and definitional word forms from an external structured lexicon. (*Right*) The encoded prefix words are given as input to the prefix function and the words in an arbitrary vocabulary are given as input to the output embedding function and the bias function to predict the next word.

tured lexicon (WordNet); we encode every word in the lexicon using its neighbors. The second follows Bahdanau et al. (2017), who used **definitions** to represent out-of-vocabulary words; we encode definitions for all words (regardless of training-set frequency).

We begin by replacing the matrix $\mathbf{E}^{in} \in \mathbb{R}^{|\mathcal{V}| \times d}$ with a neural network that defines a word's embedding compositionally from its surface form, its position relative to other words in a structured lexicon, and a natural language definition. For each word x, we refer to these, respectively, as the word type's surface embedding $\mathbf{c}_x$, relational embedding $\mathbf{r}_x$, and definitional embedding $\mathbf{d}_x$. We assume each has a dimensionality of d. The last two are optional (if missing, they are set to zero), and we redefine $\mathbf{e}_x$ as the concatenation of the three, namely $\mathbf{e}_x = \langle \mathbf{c}_x, \mathbf{r}_x, \mathbf{d}_x \rangle$. For $\mathbf{r}_x$ and $\mathbf{d}_x$, we used the structured relations (synonyms and hyponyms) and free-text definitions in WordNet (Fellbaum, 1998).

In this study, we focus on simple, computationally efficient options for the three encoders. A word x's character sequence is encoded as surface encoding $\mathbf{c}_x$ using a convolutional network followed by a highway network (Jozefowicz et al., 2016; Peters et al., 2018). Its relational encoding $\mathbf{r}_x$ is given by an average of $\mathbf{c}_{x'}$ across WordNet synonyms and hyponyms x'. The definitional encoding of x, $\mathbf{d}_x$, we similarly take an average of the surface encodings $\mathbf{c}_{x'}$ over words x' appearing in the definition. For computational efficiency, we set a maximum limit to the number of words to be used for both relations and definitions (see Appendix B.1.1). If a word's information is not in WordNet, we set $\mathbf{r}_x$ and/or $\mathbf{d}_x$ to $\mathbf{0}$. In future work, additional encodings could be added, e.g. contextualized examples (Khandelwal et al., 2020).

A notable property of these input embeddings is that their parameter count does not depend on the vocabulary size $|\mathcal{V}|$. Further, the vocabulary used in training need not be iden-

tical to the one used during finetuning, evaluation, or deployment. For example, during training we can use the full vocabulary combined with a softmax approximation method (e.g., Grave et al., 2017b), or by dynamically narrowing the choice of x_t based on its history using co-occurrence statistics (L'Hostis et al., 2016). During finetuning or evaluation, one can use the same vocabulary (required for traditional perplexity evaluations) or a different one chosen statically or dynamically, since any word's input embedding can be calculated compositionally.

4.4.2 *Compositional Output Embeddings*

One straightforward option for vocabulary size-independent output embeddings is to reuse the compositional input embeddings from Section 4.4.1, along the lines of Press and Wolf (2017). Concretely, at timestep t, we take the set $\mathcal{V}'_t$ of output word types allowed, embed each word type $v \in \mathcal{V}'_t$ as in Section 4.4.1, and stack these into a matrix $\mathbf{E}^{in}_t$ which serves directly as $\mathbf{E}^{out}$.

Though these compositional representations do enable extensive sharing across the vocabulary, we suspect that the features they capture may require additional processing before capturing "output" distributional similarity, especially when another domain is the real target use case for the language model. This follows prior work discussed in Section 4.3, which showed that making the output embedding a function of the input embeddings with shared parameters improves over simple tying.[2]

We therefore adopt a depth-k residual network for the output embedding function g (from Section 4.3) that consists of a feedforward function g_i at each layer j with d-dimensions each and apply it to the input embedding at timestep t:

$$\begin{aligned} \forall j : 1 \leq j \leq k, \mathbf{E}^{out(j)}_t &= g_j\left(\mathbf{E}^{out(j-1)}_t\right) + \mathbf{E}^{in}_t \\ \mathbf{E}^{out(0)}_t &= \mathbf{E}^{in}_t. \end{aligned} \tag{4.4}$$

Hence, we use $\mathbf{E}^{out(k)}_t$ as the output embedding at timestep t. To avoid overfitting, we apply variational dropout in between the layers, following Pappas and Henderson (2019). In contrast to that work, our resulting output embeddings are compositional. The depth k and the dropout rate are hyperparameters to be tuned on development data. The number of parameters is proportional to k times the number of parameters in the feedforward network ($O(d^2)$); it does not depend on the vocabulary size.

---

[2]Note that the input embeddings are passed through the prefix encoder f, which uses additional parameters to create the hidden state $\mathbf{h}_{t-1}$.

Bias

In conventional language models, each word in the vocabulary is assigned a bias parameter that roughly captures its log-frequency under a unigram distribution. This is the last part of a neural language model whose parameters depend on the vocabulary size. Instead of a dedicated, independent bias parameter for each word $v \in \mathcal{V}$, we define

$$b_v = \sigma\left(\mathbf{w} \cdot \mathbf{e}_v^{out} + a\right), \tag{4.5}$$

where σ is the activation function and we introduce parameters $\mathbf{w} \in \mathbb{R}^d$ and $a \in \mathbb{R}$. The bias values b_v are stacked to form $\mathbf{b}$ and used in Equation 4.3.

Training

Since all components are differentiable with respect to their parameters, the entire model can be trained to maximize training-data likelihood as described earlier (Section 4.2.1). Parameters include:

- Input character embeddings, the convolutional network for $\mathbf{c}_*$, and $3d^2$ parameters for projection (Section 4.4.1);
- Output embedding transformation, including the depth-k feedforward network for output embeddings (Section 4.4.2) and the bias parameters (Section 4.4.2); and
- Prefix encoder f, an orthogonal design choice to our method (an LSTM in our experiments).

The model size can be adjusted by changing output embedding hyperparameters to fit a given memory requirement — this is the same as any other neural network. Note that despite our vocabulary-size independent parameterization, we still need to process all the words in the supplied vocabulary leading to increased training times despite the model's sample efficiency. This can be prohibitive for very large vocabularies ($\geq 100K$), where we recommend using softmax approximation methods and making sparse updates of the output embedding parameters. During inference, $\mathbf{E}^{out}$ can be cached for fast access; there is no need to execute a forward pass more than once.

The total set of parameters for our model is the following one $\Theta = \{\Theta_{input}, \Theta_{output}, \Theta_{prefix}, \Theta_{bias}\}$. The input embedding function parameters are $|\theta_{input}| = d_{char} \times \mathcal{V}_{char} + u \times |\theta_{CNN}|$ where d_{char} is the dimension in the character lookup table in $\mathcal{V}_{char}$ and u is the number of convolutions employed. The combined form representations require only $3d \times d$ parameters that originate from their down projection. The output embedding function parameters are $|\theta_{output}| = k \times |\theta_{FF}|$, for the prefix function are $|\theta_{prefix}|$ which depends on the chosen prefix encoder function, and for the bias are $|\theta_{bias}| = d + 1$. Note that none of the above model parameters depends on the vocabulary size $|\mathcal{V}|$. In contrast to previous approaches,

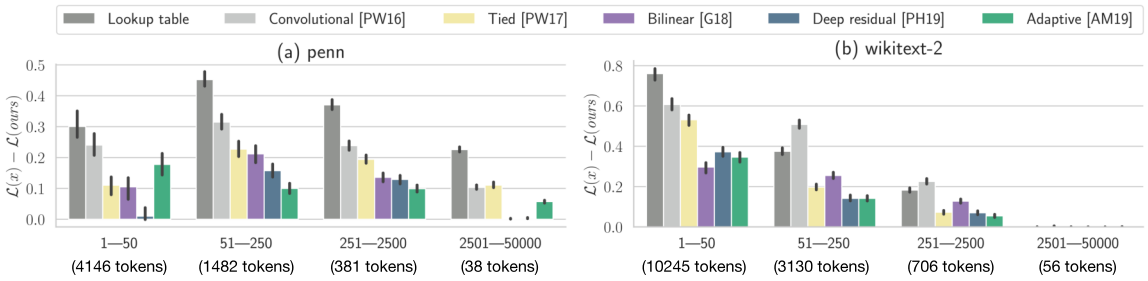

Figure 4.3: Median loss difference between each baseline and GroC over different word frequency intervals on `penn` (a) and `wikitext2` (b). The biggest differences are mostly observed on words with low training frequencies. Error bars show 95% confidence intervals for the median.

Dataset	genre	$\lvert\mathcal{V}\rvert$	# tokens	WNet cov.	
`penn`	news	10K	929K	78	86
`wikitext2`	Wiki.	33K	2M	73	76

Table 4.1: Language modeling dataset statistics. The last two columns give the percentage of the vocabulary covered by WordNet for relational and definitional encodings, respectively.

our output embeddings (i) do not require parameters that scale linearly with the vocabulary size and (ii) can be reused with any strategy for selecting the output vocabulary at inference time.

4.5 Conventional Language Modeling

We first establish the performance of GroC in the conventional closed-vocabulary setting, considering two datasets. We consider out-of-sample generalization (measured by test-set perplexity) and also analyze fit across the vocabulary by frequency bin.

4.5.1 Experimental Setup

Datasets. We evaluate our methods on two English datasets: `penn` (Marcus et al., 1993) and `wikitext2` (Merity et al., 2017). We report test perplexity using the provided training/dev./test splits. Table 4.1 also quantifies the percentage of each dataset's vocabulary that is covered by WordNet (used to derive relational and definitional encodings).

External knowledge For each of the vocabularies of the datasets we extract relational and definitional forms from Wordnet (Fellbaum, 1998). For simplicity we keep only the

words in that are present in the vocabulary itself. The coverage on `penn` is 78% and 85.8%, on `wikitext2` is 73.4% and 76.2%, on `lambada` is 48.5% and 52.9%, on `wikitext103` is 37.8% and 38.6% for relational and definitional features respectively. Unless otherwise noted, for each word we limit the number of relations to 3 and its definition length to 10.

Models.

All of the models compared use the same prefix encoder: a vanilla recurrent neural network based on the implementation by Merity et al. (2017) with 2 layers and 1024 LSTM units, regularized with hidden unit dropout of 0.65 along the lines of Grave et al. (2017a). The following output embedding approaches are compared:

- **Lookup table**: trains a full output embedding lookup table that corresponds to the vocabulary as defined in Eq. 4.3.
- **Convolutional** (Jozefowicz et al., 2016): an alternative to a lookup table that uses a character-level convolutional neural network followed by a highway network plus a linear "correction" for each vocabulary element to represent the outputs.[3]
- **Tied** (Press and Wolf, 2017): avoids training separate input and output embedding matrices by tying their parameters. This is a common technique that mitigates the overparameterization issue of the lookup table.
- **Bilinear** (Gulordava et al., 2018): performs a simple linear transformation of the input embedding to produce the output embedding that effectively shares parameters across outputs.
- **Deep residual** (Pappas and Henderson, 2019): performs a deep residual transformation of the input embedding with variational dropout in between its layers, which is more expressive than the bilinear one.
- **Adaptive** (Baevski and Auli, 2019): uses a bilinear transformation of the input and output embedding with parameters proportional to the word frequencies, to assign more capacity to frequent words and less capacity to infrequent ones. This is considered to be a state-of-the-art output embedding method.
- **GroC** (this work): the grounded compositional output embedding (Section 4.4.2).

For fair comparison, we apply variational dropout to all output embeddings. Hyperparameter selection of dropout rates, output network depth and activation, linear "correction," and adaptive frequency cutoffs was conducted by grid search on validation data.

[3]Note that we chose not to use a linear "correction" with GroC since it deviates from our goal of having a vocabulary-independent parameterization, but it could be applied to GroC in the future for additional improvements.

	penn		wikitext2	
Output embedding	$\|\Theta\|$	test	$\|\Theta\|$	test
Lookup table	13M	90.8	23M	108.3
Convolutional [J16]	13M	101.6	23M	116.6
Tied [PW17]	10M	86.2	15M	97.3
Bilinear [G18]	10M	83.7	15M	95.9
Deep residual [PH19]	10M	80.5	15M	94.7
Adaptive [AM19]	8M	79.3	9M	90.7
GroC (ours)	9M	69.5	9M	82.5

Table 4.2: Perplexity scores on conventional language modeling benchmarks with closed vocabulary. $|\Theta|$ denotes the total number of model parameters.

4.5.2 Results

Table 4.2 reports perplexities achieved by all seven models. The main finding is that GroC achieves lower perplexity than the previous models, on both datasets. Note that GroC outperforms the state-of-the-art output embedding method of Baevski and Auli (2019); specifically, by -9.8 and -8.2 points on `penn` and `wikitext-2`, respectively. The difference with the other methods is even larger. We also confirm the findings of Pappas and Henderson (2019), that output parameter sharing methods outperform tied output embedding and the lookup table, and of Jozefowicz et al. (2016), that convolutional output embeddings lag behind full softmax (lookup table). Notably, GroC outperforms the best reported scores by Merity et al. (2017) and Grave et al. (2017a) on `penn`, using about 11M fewer parameters and a similar prefix network to the latter.

Nevertheless, GroC is about $1.3\times$ slower than the convolutional method on `penn`; with sparse updates ($p > 0.3$) we can make it $2.1\times$ faster than that method, which is comparable to the speed of the bilinear method, while maintaining a perplexity improvement of -26 points.

4.5.3 Analysis

The experiment above establishes that our approach achieves improved perplexity relative to alternative output embeddings. We next decompose its performance in various ways to understand why.

Word frequency effects. We conjecture that GroC's main benefit comes from words that are rare in the training data, since the core contribution is to share representations across the vocabulary. To evaluate this hypothesis, we consider the difference in test loss (cross entropy) between GroC and a baseline model, following Baevski and Auli (2019) but computing the median instead of the average to reduce the effect of outliers. We decompose

	penn		wikitext2	
Model	dev.	test	dev.	test
GroC + out	75.0	71.4 -	87.0	82.5 -
– relations	77.1	72.7 ↑	90.2	85.3 ↑
– definitions	75.6	72.0 ↑	88.6	84.3 ↑
– both	79.8	75.8 ↑	94.3	89.8 ↑
GroC	72.5	69.5 -	88.7	84.1 -
– relations	74.2	70.8 ↑	93.0	88.0 ↑
– definitions	74.4	71.1 ↑	87.6	83.1 ↓
– both	76.3	73.2 ↑	94.5	89.5 ↑

Table 4.3: Ablated model variants on `penn` and `wikitext-2`. *out*: the deep residual output network.

Coverage	surf.	0%	16%	32%	48%	64%	82%
inference	73.1	187.8	159.3	128.5	102.6	83.5	69.5
train	–	72.4	70.0	70.4	69.6	70.7	69.5

Table 4.4: External lexicon coverage effect on the perplexity of GroC on the `penn` test set. *surf.*: model with surface forms only from Table 4.3, last row.

this score by data frequency bins (e.g., words occuring 1–50 times in the training dataset). Figure 4.3 is displayed for the `penn` and `wikitext2` datasets. The trend we observe is that GroC has the greatest benefit for words in lower frequency bins, relative to each model. The lowest-frequency bin on `penn` deviates from this pattern, which we take as an indication that generalizing to infrequent words with only 1M training tokens and a small 10K vocabulary is inherently challenging.

Ablations. To assess the contributions of GroC's components, we performed ablation tests on `penn` and `wikitext2` (Table 4.3). These include removing relational and/or definitional forms, either with or without a deep residual output network. For fairness, we tune the hyperparameters of the ablated model variants as above. Overall, removing the relational and definitional forms from the main model with or without output network on top increases the perplexity. The largest drop in perplexity happens when we remove both forms, which highlights their notable contribution to the full model. Lastly, the results on `wikitext2` highlight the importance of capturing the output similarity with an output network (out) for datasets with a larger vocabulary as opposed to merely reusing the grounded compositional embeddings as output embeddings.

Lexicon coverage. To measure the effect of lexicon coverage on model performance in a controlled setting, we artificially remove words from WordNet, making them unavailable

Dataset	source	train $\|V\|$	test $\|V\|$	OOV%
2007	News Crawl	81K	188K	2.0
2008		82K	197K	2.3
2009		81K	195K	2.5
2010		78K	181K	2.4
2011		80K	184K	2.5
web	Common Crawl	75K	174K	5.8
wiki	WikiText-103	67K	109K	5.4

Table 4.5: Dataset statistics for cross-domain experiments. OOV% gives the percentage of tokens in the test set not present in the 2007 train vocabulary.

for relational and definitional encodings. In this experiment, we consider the penn dataset, where WordNet's coverage over the (relatively small) vocabulary is highest to begin with. Table 4.4 shows the resulting test perplexity of a pretrained model (inference) and a model trained from scratch (train) when such controlled manipulation is applied to them from 0% up to the maximum of 82% coverage (Table 4.1) . Note that we treat relational independently of definitional forms since they are not always co-present. Overall, the results indicate that the model is sensitive to changes in the forms of words that have been seen during training but it is robust to changes if it is trained from scratch. In the next section, we investigate what happens when we add forms for words which the model has never seen before.

4.6 Cross-Domain Language Modeling

To demonstrate our model's ability to generalize beyond its training data, we evaluate it across domains with an open vocabulary, in two settings: zero-resource, where it is first trained on one domain and then tested on a new target domain, and low-resource, in which the model is further exposed to training data in the new domain.

4.6.1 Experimental Setup

Data. Following Grave et al. (2017a), we create English datasets from News Crawl (Bojar et al., 2014), Common Crawl,[4] and WikiText-103 (Merity et al., 2017). Dataset statistics are given in Table 4.5. All models are trained on 2M tokens from the 2007 dataset and evaluated on 10M tokens; finetuning is done on an additional 2M tokens from the target domain. We consider the domain of the 2008–2011 datasets to be similar ("near") to that of the training set, 2007, as they contain news from different time periods. In comparison, web and wiki are more different ("far") from 2007.

[4]We used the version from WMT 2014 (Bojar et al., 2014).

Models. We compare GroC to the tied output embedding model described in Section 4.5.1 when combined with the following adaptation methods:

- **Unigram**: we interpolate the model's distribution with a unigram cache, which assigns probabilities based on the counts of words in the test data observed so far during evaluation.
- **Local cache**: we interpolate the model's distribution with a neural cache (Grave et al., 2017c), which assigns probabilities based on the similarity of the current hidden state to previous hidden states during evaluation.
- **Finetuning**: the model is finetuned on 2M tokens from the target domain.

(We also compare to the reported unbounded cache results from Grave et al., 2017a.)

Cache models provide effective adaptation without training by using recent history to develop an auxiliary distribution during evaluation, informing predictions of unseen or rarely-seen words. However, as GroC already assigns non-negligible weight to new words not seen prior to evaluation, the cache has less effect by default, even if its predictions are more accurate, an effect we observed in validation. To address this, we down-weighted the model's predictions for new words prior to cache interpolation by 0.1, a weight selected on the `wiki` validation set. Cache hyperparameters (interpolation weight and flattening weight) were tuned with the tied model on the `2008` validation set. For finetuning, both tied and GroC models were trained for an additional 3 epochs on the target domain, allowing them to adapt to the new domain. See Appendix B.2.4 for hyperparameter details.

Vocabulary setting. For a fair comparison, all models are evaluated on the union of the training and test vocabularies. Tied models are interpolated with the uniform distribution at test time to prevent infinite perplexities on unseen words, prior to cache interpolation if applicable. Words present in the finetuning data but not in the original training data are given random embeddings prior to finetuning.

4.6.2 Results

The results for the cross-domain experiments are shown in Table 4.6. Standalone GroC improves perplexity relative to the tied model in every domain by up to -30 points, including the local neural cache and the unbounded neural cache model in the near-domain, even when the former is applied to our own stronger tied-embedding baseline model. In addition, when finetuned on the target domain GroC outperforms all non-finetuned baselines by a wide margin including the unbounded cache by about -40 and -132 points on near and far domains, respectively, indicating that even a small exposure to target domain data dramatically improves generalization. Finetuned GroC also outperforms the

	Model	2007 →	near domains 2008	2009	2010	2011	far domains Web	Wiki
	Zero-resource setting							
Grave et al. (2017a)	Base	220.9	237.6	256.2	259.7	268.8	689.3	1003.2
	Base + unigram	220.3	235.9	252.6	256.1	264.3	581.1	609.4
	Base + local cache	218.9	234.5	250.5	256.2	265.2	593.4	316.5
	Base + unbounded cache	166.5	191.4	202.6	204.8	214.3	383.4	337.4
Baseline	Tied	184.3	199.8	217.3	221.6	229.9	660.6	841.1
	Tied + unigram [G17]	187.8	203.6	221.5	225.9	234.3	577.5	819.7
	Tied + local cache [G17]	181.8	196.5	212.0	217.7	225.9	501.7	**406.9**
Ours	GroC	158.6	171.0	186.7	192.5	200.4	637.9	753.9
	GroC + unigram [G17]	155.2	167.3	183.1	189.5	196.4	533.6	689.2
	GroC + local cache [G17]	**152.6**	**164.1**	**179.0**	**185.1**	**192.3**	**493.0**	408.8
	Low-resource setting							
	Tied + finetuning	–	172.8	177.9	180.7	185.4	**212.7**	242.6
	GroC + finetuning	–	**153.7**	**162.2**	**167.0**	**170.6**	239.5	**216.9**

Table 4.6: Results on *near* and *far* cross-domain language modeling with an open vocabulary with a zero-resource or a low-resource setting. Top four rows display scores from Grave et al. (2017a), while the next three are from our re-implementation with a stronger base model. Boldface marks the best perplexity on each test set within each setting (zero- or low-resource).

finetuned tied model by up to -25 points except in `web` domain, and reaches lower validation scores with fewer iterations in 5 out of 6 domains (see Appendix B.2.1). For the `web` domain, caches and finetuning are more effective than in any other domain, indicating unique domain dynamics that may impact the effectiveness of GroC-based adaptation.

4.7 Summary

In this chapter, we described an adaptive language model based on grounded compositional outputs, drawing from language-specific lexical resources (WordNet) as well as information contained in the text (a word's surface form). We demonstrated that this model reduces the number of parameters and increases sample efficiency, outperforming existing strong output embedding methods and adaptation baselines on in-domain and open-vocabulary settings respectively. In principle, our results should also be applicable to wordpiece language models which are currently based on lookup tables to improve their sample efficiency and compactness. Future extensions of this work could investigate to what extent pretrained language models benefit from GroC on such zero-resource or low-resource adaptation settings. This work suggests several possible future directions for language modeling in low-resource domains: scaling training to even larger vocabularies, applying GroC in a large pretraining setting to expand its zero-shot generalization, and extension to other languages, the last of which we will examine in Chapter 5.

Chapter 5

POLYGLOT COMPOSITIONAL OUTPUT EMBEDDINGS

5.1 Introduction

Large language models (Devlin et al., 2019; Radford et al., 2019, inter alia) have been highly effective in multilingual NLP, allowing crosslingual sharing of both annotated and unannotated information (K et al., 2020; Dufter and Schütze, 2020; Pires et al., 2019). However, these models demand significant amounts of pretraining data, performing worse for lower-resource languages due to impoverished target language lexicons and undertrained representations (Chau et al., 2020). The research described in Chapter 4, examining sample-efficient word-level language modeling, demonstrated effective domain adaptation with little or no target domain data, by making use of compositional input and output representations, but considered this approach only within a single language, English.

In this chapter, we examine whether such an approach can be successfully applied to *crosslingual* transfer via joint training in a low-resource setting. We train several small LSTM language models on small amounts of text in 10 languages, and compare monolingual models to those trained on two languages jointly. We find that joint multilingual training consistently improves the perplexity of baseline models, regardless of the embedding type, and that models using compositional output embeddings continue to have the best performance overall. This suggests that the benefits of joint multilingual training do not rely on word-specific parameters, and that shared linguistic features can be encoded in compositional embedding networks.

Finally, we study whether careful initialization of the model can make training more efficient for multilingual compositional LMs, compared to training from a baseline of random initialization. First, we train a model with compositional *inputs* only, and traditional output embeddings (comparable to ELMo or Rosita), on multiple languages. At test time, the compositional embedding network is then used to compute output embeddings as well, replacing the lookup output embeddings (Section 5.5). This avoids the computational expense of training a model with compositional output embeddings, while still having the benefit of being able to handle OOV words at test time. However, we find that this model does not outperform multilingual GroC trained from scratch with output embeddings, even after finetuning.

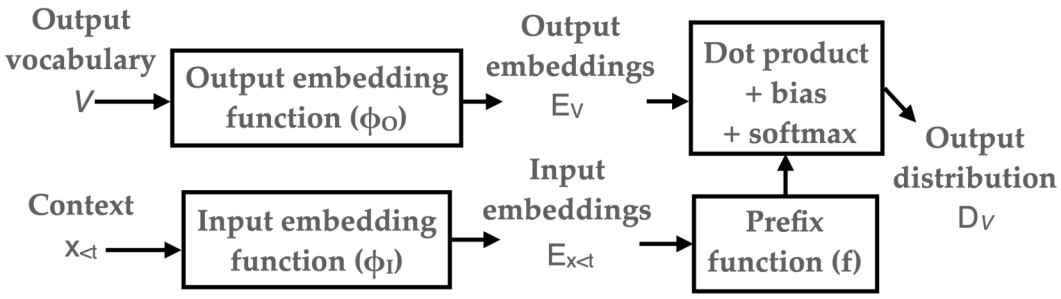

Figure 5.1: A generalized language model. Input and output words are encoded by embedding functions; the input embeddings are fed to a prefix function (e.g., an LSTM or transformer), and the output (the hidden state) compared to each of the output embeddings to form a probability distribution over the vocabulary. The input and output embedding functions ϕ_I and ϕ_O may share parameters (*tied* embeddings).

5.2 Language Model Embeddings

Language modeling is the task of assigning probabilities to sequences of tokens representing natural language, which is often implemented by finding the conditional probability of each token given the sequence of previous tokens:

$$p(\boldsymbol{x}) = \prod_{t=1}^{T} p(x_t \mid x_1, \ldots, x_{t-1}). \tag{5.1}$$

As discussed in Section 4.2, neural language models can generally be decomposed into three main parts: an input embedding function, which produces a representation $E_{x_{<t}}$ of each word or token in the context $x_{<t}$; a prefix function, which combines those representations to produce a hidden state or context vector $h_{<t}$; and an output embedding function, which produces a representation for each word in the vocabulary E_V (see Figure 5.1. The next word distribution is then given by

$$p(X_t = x_t \mid h_{<t}) = \text{softmax}(E_V h_{<t} + b) \tag{5.2}$$

where b is a bias vector.

Most commonly, the input and output embedding functions consist of simply selecting the vector corresponding to the desired vocabulary element from an embedding matrix (a *lookup embedding*). This matrix is often shared between input and output embeddings (Press and Wolf, 2017; Inan et al., 2017). Other parameterizations include adaptive embeddings (Grave et al., 2017b; Baevski and Auli, 2019), hierarchical softmax (Morin and Bengio,

2005) or compositional functions to produce embeddings, (e.g. Jozefowicz et al., 2016). In ELMo (Peters et al., 2018) and in the multilingual language models described in Chapter 3, the input embedding is a convolutional neural network over the sequence of bytes (or characters) in each word, while the output embedding is a lookup embedding. In GroC (see Chapter 4), both input and output embeddings are implemented by the compositional function shown in Figure 4.2, which includes an ELMo-like CNN; the same parameters are used to embed both context and vocabulary, and are updated via gradients from both paths of the computation graph. This results in better representations for rare words, at the cost of higher memory usage and slower training (since the output embedding must be frequently recomputed, consuming memory for gradients as well as the final vector). While GroC is not an ideal approach for large models due to these memory and computational requirements, it is well suited to answering our research question of whether dedicated parameters for individual embeddings are a key component of crosslingual transfer.

In this chapter, we use LSTM language models based on the AWD-LSTM codebase[1]; i.e., the prefix function is a 2-layer LSTM for all models, while the input and output embedding functions vary. While Transformer language models (Vaswani et al., 2017) have outperformed LSTMs at large scales, in this work we examine only very small datasets, where the performance gap is small and dependent on optimization (Dai et al., 2019). For this reason, we adapt the code from Chapter 4 (as published in Pappas et al., 2020) to a multilingual setting while carrying over most of the same hyperparameter settings for the Penn Treebank experiments in that chapter, as that dataset is of a similar size to the ones used here.

5.3 Data

In this work, we consider word-level language modeling with small datasets. As shown in Chapter 4, GroC's sample-efficient architecture allows strong performance in monolingual modeling at small data scales; we wish to show whether this means that multilingual GroC will enable crosslingual transfer at small data scales.

We use text from the `wiki40b` corpus (Guo et al., 2020), a multilingual corpus of Wikipedia text filtered to remove non-content sections (such as References and See Also) and structured content (figure captions, tables and lists). This is intended to produce a sample of natural language which is relatively stylistically uniform across the chosen languages. For each language, we sample approximately 1M words of training text and 100k words for development and test sets, drawn from the existing splits in `wiki40b`, segmented into words in a language-specific manner with the Stanza pipeline (Qi et al., 2020). We study two types of crosslingual transfer: joint training with a target language and English, and joint training with a target language and a related language. Languages, grouped with their related language, and dataset statistics are shown in Table 5.1.

[1] github.com/salesforce/awd-lstm-lm

Lang	Code	$\|V\|$	Family
English	en	69955	Indo-European
French	fr	79905	Indo-European
German	de	124920	West Germanic
Dutch	nl	85323	West Germanic
Italian	it	89577	Italo-Western Romance
Spanish	es	87107	Italo-Western Romance
Indonesian	id	82022	Malayo-Polynesian
Tagalog	tl	86898	Malayo-Polynesian
Ukranian	uk	149658	East Slavic
Russian	ru	154198	East Slavic

Table 5.1: List of the languages and the statistics of the sampled data used in our experiments. Each shaded/unshaded section corresponds to a pair of *related* languages; the closest language family they share according to the Ethnologue phylogenetic tree (Eberhard et al., 2021) is given in the last column. English and French are treated as a related language pair due to strong vocabulary influence (Millward and Hayes, 2012) despite their phylogenetic distance. Vocabulary size is the count of distinct word types after segmentation with the Stanza pipeline (Qi et al., 2020).

After word segmentation, we construct the vocabulary for a given language as the set of all word types appearing in the training and development sets. Our selection focuses on languages with Latin and Cyrillic scripts; these languages have a balance between high character-word ratios, such that a compositional embedding enables significant sharing between words, and limited word-level vocabularies (a few hundred thousand words at our data scale), which can fit in GPU memory despite GroC's memory-intensive design. For example, the specific architecture of GroC is probably not a good choice for Chinese due to low compositionality, or Turkish due to the intractability of word-level modeling in an agglutinative language. We leave investigation of compositionality in logographic scripts, or in combination with subwords, to future work.

We use the Open Multilingual Wordnet (Bond and Foster, 2013) as the lexicon to construct the relational and definitional features in GroC models. However, of the ten selected languages, only five (English, French, Dutch, Italian and Spanish) appear in the Open Multilingual Wordnet. Furthermore, the non-English portions largely consist of links from non-English lemmas to synsets from the original English WordNet, and thus all definitions appear only in English, requiring a model trained on English text to properly construct definitional forms for any language. Thus in this work, we only evaluate models with relational and definitional forms for transfer between English and one of the other four languages. We leave exploration of lexicon-grounded compositional embeddings in addi-

tional settings to future work.

5.4 Joint Multilingual Training With Output Composition

As we have seen in earlier chapters, crosslingual joint training and compositional architectures are each useful for low-resource learning. If the two techniques can be applied together, and have an additive benefit, then this would enable stronger models for low-resource languages with less target language data. However, lookup embeddings are the dominant paradigm in large multilingual language models such as mBERT (Devlin et al., 2019) and mT5 (Xue et al., 2021), and while the multilingual ELMo model described in Chapter 3 used compositional input embeddings, it still relied on compositional output embeddings during pretraining.

To determine whether these two techniques can be used together, we extend Chapter 4's compositional word-level language models to a multilingual setting, to study models without dedicated vocabulary parameters. Following that work, we focus on word-level language modeling, instead of using subwords or character-level prediction. We construct the training vocabulary as the union of all word types in the training sets of each language. If the same word (i.e., the same character sequence) appears in multiple languages, by default it is treated as the same vocabulary item in all models: in models with embeddings, it receives a single embedding; in models with compositional representations, the compositional representation is computed in the same way regardless of the language context, including using a single definition and set of relations. We choose this handling of words that appear in multiple languages because Open Multilingual WordNet statistics indicate that in almost all cases, they have very similar meanings in each language. For example, about 15% of words in the English vocabulary also appear in the Spanish vocabulary, including punctuation, names, abbreviations, quotations of text in the opposite language, etc. Of those, for only 5.2% is the English definition used not a correct definition for Spanish (i.e., the word has an entry in the Spanish WordNet and the selected English synset is not linked to the Spanish lemma), representing only 0.8% of the full vocabulary. Similar proportions hold for the other language pairs in OMW. Thus, treating these words as a single vocabulary item is correct in the overwhelming majority of cases. Future work could apply a more nuanced approach, perhaps sharing some forms but not others, or sharing only for certain words.

To handle a multilingual vocabulary, including language pairs with different scripts, we compute the surface form with a convnet over Unicode characters, rather than over bytes as in Chapter 4, similar to the modification we made to the ELMo architecture in Chapter 3. This increases the size of the character vocabulary, but allows dedicated parameters for characters in different scripts.

tgt lang	lookup (mono)	lookup (EN+tgt)	lookup (rel+tgt)	ELMo (mono)	ELMo (EN+tgt)	ELMo (rel+tgt)	GroC (mono)	GroC (EN+tgt)	GroC +lex (EN+tgt)	GroC (rel+tgt)
en	364	-	*340*	275	-	276	**269**	-	-	270
fr	274	*239*	*239*	197	*194*	*194*	186	*181*	**173**	*181*
de	690	*627*	*600*	417	419	*397*	412	*403*	-	**391**
nl	388	*355*	*345*	266	*261*	*252*	257	*253*	**250**	**250**
es	345	*292*	*312*	236	*232*	*220*	211	*206*	215	**203**
it	481	*435*	*437*	328	*323*	*305*	298	299	*293*	**282**
id	672	*642*	*579*	477	*442*	512	438	**430**	-	444
tl	239	*215*	*230*	171	*159*	179	161	**154**	-	*157*
uk	1293	*1252*	*1162*	702	*679*	*632*	549	*534*	-	**469**
ru	1300	*1227*	*1235*	732	*701*	*641*	544	571	-	**480**
Avg.	631	*587*	*571*	392	*379*	*370*	340	*339*	-	**317**

Table 5.2: Perplexity for models trained on 1M tokens per language of `wiki40b`. Monolingual models ("mono") train only on the target language. Multilingual (polyglot) models train jointly on the target language combined with English ("EN+tgt") or with the related language given in Table 5.1 ("rel+tgt"). **Bold** indicates the best model for a language pair; *italics* indicates a polyglot model that improves relative to the monolingual model of the same type. "Avg." is over non-English target languages only.

Memory optimizations The union of the vocabularies of N languages scales approximately linearly with N. GroC is already a memory-intensive embedding method, because the embedding network is required to generate the embedding for every word in the output vocabulary, with memory requirements of $|V|$ times the size of all the gradients in the embedding network. Thus, to fit polyglot models with large vocabularies in memory, we use monolingual batches and compute the output embeddings only for the words in the vocabulary of the current batch's language. This *language-screened softmax* is not used in prior work on jointly trained polyglot language models, so we compare the perplexity of lookup models trained with and without language-screening, to show that they perform roughly equivalently. See Appendix C.

5.4.1 Comparison to baselines

We first compare GroC to lookup models and ELMo-like models in both monolingual and multilingual settings, to determine whether the perplexity improvement from polyglot training which we see for lookup models also applies to models with compositional embeddings. With this experiment, we can determine whether GroC is benefiting from crosslingual transfer in the same way as non-compositional language models.

Table 5.2 shows perplexity for monolingual and multilingual models of three types.

- Lookup: the embedding function is simply a matrix with vector embeddings for each

word in the vocabulary, shared between input and output. This is most similar to traditional LSTM language models such as Merity et al. (2018). For fair comparison to other variants, which are all word-level models, each embedding corresponds to a word type in the vocabulary (i.e., not BPE, wordpieces, characters).

- ELMo: the input embedding function consists of the compositional embedding function from Chapter 4, shown in Figure 4.2, while the output embedding function is an embedding matrix. This model is similar to ELMo (Peters et al., 2018) and Rosita (Chapter 3), except that those works used a bidirectional LSTM to provide full-sentence context, while our prefix function is unidirectional to compute valid perplexities.

- GroC: a shared compositional embedding function is used for both input and output embeddings. For languages present in the Open Multilingual Wordnet, we include a comparison to a model with the relational and definitional forms described in Chapter 4 ("GroC+lex"), which incorporates English definitions for target language words and crosslingual relations. For all other GroC models, only the surface form is used.

For each combination of target language and model type, we train a monolingual model in that language, a polyglot English+target model, and a polyglot related+target model, and compare the perplexity on the target language's test set. In each of these models, the prefix function is the same; only the input and output embedding functions vary.

As seen in Table 5.2, polyglot training tends to improve performance for all models. We also see that, consistently across all model types, sharing between related languages tends to provide a larger benefit than sharing with English. This effect is not seen in the compositional models for Indonesian and Tagalog, notably more distantly related than any other related-language pair except French and English; each of them instead benefits more from transfer with English. However, when averaged across languages the improvement from related-language transfer is large. We also see that compositionality is helpful regardless of the language; ELMo models consistently outperform lookup models, and GroC models consistently outperform ELMo. Thus, the best model for almost every language, as well as on average, is GroC model trained on multiple languages.

We also find that the relational and definitional forms ("+lex"), while shown to be helpful monolingually in Chapter 4, are only inconsistently useful for crosslingual transfer. Adding these forms helps transfer from English in French, Dutch, and Italian, but the gain is small, and perplexity increases in the case of Spanish. This effect may be due to poor coverage in non-English languages; further research with less-rich but higher-coverage resources such a bilingual dictionaries might uncover better forms of crosslingual supervision.

tgt lang	lookup 2M (mono)	lookup 2M (EN+tgt)	lookup 2M (rel+tgt)	GroC 1M (mono)	GroC 1M (EN+tgt)	GroC 1M (rel+tgt)
en	260	-	**228**	*269*	-	270
fr	197	**174**	**174**	186	*179*	*179*
de	489	*414*	*397*	412	*403*	**391**
nl	281	*235*	*251*	257	*253*	**251**
es	251	*223*	*225*	211	*206*	**203**
it	320	*282*	*283*	298	299	**282**
id	471	*430*	**418**	438	*430*	444
tl	173	*149*	**153**	161	*154*	*157*
uk	816	*718*	*700*	549	561	**469**
ru	843	*712*	*702*	544	571	**480**
Avg.	427	*366*	*367*	340	*339*	**317**

Table 5.3: Perplexity for lookup models trained on 2M tokens per language of wiki40b. GroC results are copied from Table 5.2 for ease of comparison.

5.4.2 Comparison across data scales

The polyglot improvements shown in Table 5.2 are smaller, when measured either as an absolute reduction in perplexity or as a percentage reduction in relative perplexity (averaged across languages), for the ELMo-like and GroC models than for the lookup models. The meaning of this difference is not clear; it may be either that compositional embeddings are less suited to crosslingual joint training or that the models with compositional embeddings are already performing well enough that there is less potential benefit from crosslingual joint training. To help disambiguate this effect, we train several lookup models using twice as much training data in each language, 2 million words, and evaluate them on the same test sets as used in Table 5.2. For ease of comparison, we do not change the configuration of the model from that used for the 1M-scale data. There are several advantages of more data, even without scaling the model: embeddings for some words will be updated more frequently, embeddings for some words not present in the original 1M-scale training set will now be learned, allowing the model to predict them at test time, and the parameters of the prefix function will all be trained on twice as many contexts, reducing overfitting. Note also that the polyglot lookup models receive twice as much text from the added language, increasing the advantage from polyglot training. Table 5.3 shows the results, including the original 1M-scale GroC models for comparison. We see that GroC usually performs about as well or better than lookup models trained on twice as much data, retaining a significant advantage when averaged across languages. However, when comparing the monolingual and polyglot averages within each model type, the im-

tgt lang	tgt↔EN (lookup)	tgt↔EN (ELMo)	tgt↔EN (GroC)
fr	34.3	53.6	52.9
de	26.6	33.4	31.9
nl	33.5	50.2	47.2
es	28.1	46.1	47.1
it	30.4	51.3	49.7
id	37.4	50.1	48.0
tl	54.3	61.5	58.3
uk	2.8	3.4	3.4
ru	2.9	2.3	2.7

Table 5.4: Alignment scores between English and target language embeddings (p@5; higher is better). Embeddings for each target language are drawn from the polyglot (EN-tgt) models and aligned to English embeddings from the same model. Alignments are learned with supervised MUSE with 10 iterations of refinement and evaluated with the accompanying dictionaries and the CSLS distance metric (Conneau et al., 2018).

provement from polyglot training is still somewhat smaller for GroC. This suggests that while compositional models do benefit from polyglot training, the lack of word-specific parameters prevents them from improving to the same degree.

5.4.3 Embedding spaces

To study why models with compositional embeddings improve less consistently from crosslingual training, we study the alignment of the monolingual models' input embeddings with MUSE (Conneau et al., 2018). We use the MUSE dictionaries to learn and evaluate supervised alignments between English and each target language. Given that alignment scores reflect similarity of embedding spaces across languages, we hypothesize that models that benefit most from joint crosslingual training will also see higher alignment scores, if the benefit of joint crosslingual training comes from being able to represent different languages in a shared space.

Table 5.4 shows the results. Somewhat contrary to our hypothesis, the output embeddings of polyglot ELMo and GroC models result in better alignments, even though those models saw smaller improvements in perplexity from joint crosslingual training. This may reflect overall embedding quality. Embeddings for lookup models tend to obtain worse monolingual word similarity scores than compositional models (ELMo and GroC) when evaluated on the SEMEVAL17 data (Camacho-Collados et al., 2017) for German, Spanish, and Italian (Table 5.5), suggesting that lookup models' embeddings are too noisy to align when learned at this low-resource scale. This interpretation is also consistent with the per-

tgt lang	covered words	lookup	ELMo	GroC
de	263/500	0.08	0.01	0.21
es	289/500	0.04	0.17	0.37
it	291/500	0.07	0.16	0.29

Table 5.5: Monolingual word similarity results. Score is rho similarity (higher is better) as evaluated by MUSE, before alignment. The same polyglot models (EN-tgt) as in Table 5.4 are used. "Covered words" indicates how many of the words in the evaluation dataset are found in the model vocabulary; while compositional models could generate embeddings for missing words, for a fair comparison we use the same vocabulary for all models in this experiment.

plexity results shown in Table 5.2, where compositional models obtain better perplexity scores than lookup models overall, presumably due to higher-quality word representations.

5.5 Initialization from ELMo-like Models

In this section, we examine whether the compositional embedding networks in models trained with lookup output embeddings (e.g., ELMo) are also useful as output embedding networks—essentially, whether the same information is learned by input and output compositional embeddings. To study this, we use compositional input & lookup output models to initialize GroC models, and examine the effect on language modeling performance and total training time. Specifically, we take the convolutional surface form networks used as *input* embeddings in the pretrained multilingual ELMo-like models in Section 5.4 and replace those models' original *output* embeddings with them. This results in a GroC-like model without the need for pretraining with a compositional output embedding (which is comparatively slow and memory-intensive).

The GroC models in Section 5.4 were trained using the same parameters for input and output embedding; thus the resulting input and output embedding spaces are identical. Because the ELMo models use different parameters for input and output embeddings, the embedding spaces are not identical, nor even necessarily aligned. This means that simply using an ELMo model's compositional input embedding for the output is unlikely to work well without finetuning, as the LSTM parameters will be adapted to map from the input embedding space to the original output embedding space. However, we hypothesize that the amount of finetuning required may be small enough to significantly shorten the total training time compared to training a GroC model from scratch.

The results in Table 5.6 bear out these hypotheses. Without finetuning, the models'

tgt lang	starter (ELMo)	equiv. GroC	random (no train)	ELMo → GroC (no ft)	ELMo → GroC (ft 1 epoch)	ELMo → GroC (ft 5 epoch)	ELMo → GroC (ft 40 epochs)
fr	194	179	226214	1.55×10^{19}	313	234	**176**
de	419	403	475116	6.21×10^{31}	799	518	432
nl	261	260	122231	6.93×10^{17}	445	392	**252**
es	232	206	154192	1.31×10^{23}	360	285	**205**
it	323	299	167018	1.97×10^{18}	565	446	**285**
id	442	430	146486	3.61×10^{20}	851	572	433
tl	159	154	251314	6.67×10^{15}	237	186	**146**
uk	679	561	248846	5.89×10^{18}	1149	801	**530**
ru	701	571	267432	6.52×10^{29}	1268	826	**567**
Avg.	379	340	228761	6.97×10^{30}	665	473	**336**

Table 5.6: Perplexity for models initialized from the input embedding of an ELMo model. "Starter (ELMo)" is the perplexity of the original model (EN-tgt polyglot), and "equiv. (GroC)" is the corresponding GroC model for that language pair (also shown in Table 5.2). "Random (no train)" is the perplexity of a randomly initialized model without training. "(no ft)" indicates the model initialized based on the ELMo model but not finetuned after, while "(ft)" indicates the model was finetuned for the stated number of epochs with an initial learning rate of 0.001. Bold indicates models that surpass the corresponding GroC model trained from scratch for 100 epochs.

perplexity is dramatically worse than even untrained models, reflecting the misalignment between input and output embedding spaces in the original ELMo-like model. The prefix function's output is optimized for the old embedding space, and the new output embedding space (without finetuning) is highly structured, yet inconsistent with the old one; as a result, rather than just selecting words from the vocabulary at random, the model is anti-optimized to predict consistently-wrong words. However, after finetuning for only 1 epoch, the performance is in the range of an equivalent lookup model, and in only 40 epochs, compared to 100 for the models trained from scratch, the finetuned models consistently surpass the performance of the ELMo models used to initialize them and frequently surpass the performance of a GroC model trained from scratch for the same language pair (compare to Table 5.2). This suggests a productive direction for future work would be to further optimize the initialization procedure, perhaps by using an auxiliary loss to keep the input and output embeddings of the ELMo model aligned, to obtain high-performing GroC models at even lower computational cost. See Appendix C for details of the development experiments used to determine the finetuning procedure used in this section.

5.6 Related Work

Edunov et al. (2019) use the input representations from pretrained ELMo models in an encoder-decoder machine translation model and find that they are effective only in the encoder, and hurt performance when used as the target-language inputs to the decoder. Our results suggest that compositional representations can be useful when decoding for language modeling, even when originally trained as input representations, but require direct finetuning of the representations; further study on initialization from ELMo models may be useful to explain this difference. For example, finetuning only the contextualizer, or freezing the parameters learned as part of the ELMo model and learning a linear transformation between the contextualizer hidden state and the output embedding, could determine whether finetuning is needed to add new information to the embedding function, or if it only corrects the alignment between hidden space and embedding space.

In this work, we do not compare to subword embedding methods such as BPE (Sennrich et al., 2016) or unigram language model encoding (Kudo, 2018), which are commonly used in large language models. While these methods address some of the same concerns as GroC, subword methods are primarily motivated by computational concerns: minimizing the vocabulary size and sequence length for more efficient training. They do not provide a means for the incorporation of monolingual or crosslingual lexical database features, other forms of crosslingual supervision or word-level evaluation, which would make direct comparison to models using other embedding methods (with different vocabularies) possible. Algorithms that determine a vocabulary from the training data may also be more vulnerable to domain transfer issues when applied at very small scales where the proportions of character patterns in the training data may not be representative of the unseen vocabulary.

Nevertheless, subword lookup embeddings are known to be a highly successful method when applied at scale, and future work considering them in a low-resource context could help with the application of insights from large language models to the low-resource context.

5.7 Summary

In this chapter, we applied GroC, the compositional output language model from Chapter 4, to a multilingual setting, and showed that it is an effective architecture for low-resource language modeling whether trained monolingually or crosslingually. We found that while crosslingual lexicons provided only minor improvements, purely character-based word representations were still effective for crosslingual modeling, consistent with our earlier results in Chapter 3. We also showed that GroC models learn high-quality embeddings that achieve high crosslingual alignment scores, allowing the creation of vocabulary-independent multilingual embedding spaces. Finally, we found that quick-training compositional *input* embeddings learn much of the same information as compositional output embeddings, allowing faster production of GroC models through a two-stage training procedure. These results demonstrate the usefulness of both compositional and polyglot models.

One finding of this chapter is that lexical features such as related words and definitions were less useful in the crosslingual setting than when applied monolingually in Chapter 4. One possible reason is noise due to poor matching of word sense between languages. A more careful approach, incorporating richer information such as part-of-speech and contextual word sense disambiguation, might be able to improve modeling of low-resource languages without requiring more text. On the other hand, the use of crosslingual lexicons was also limited by the availability of such lexicons for only a few of our languages of interest, and limited vocabulary coverage within those languages. Future work could instead make use of less expensive resources such as bilingual dictionaries as proxies for crosslingually related words.

Chapter 6

CONCLUSION

In this dissertation, we have improved models of natural language in low-resource settings through crosslingual joint training (polyglot modeling) and vocabulary-independent parameterizations. Our results showed that each of these approaches, as well as their combination, are promising for flexible and adaptable low-resource models of natural language. This work suggests several directions for future research.

6.1 Contributions

In Part I, we presented research that takes a "polyglot training" approach to multilinguality, in which models are trained on data from multiple languages. We used this approach for both supervised models and language models, and evaluated it on several structured prediction tasks. Chapter 2 presented a semantic role labeling model trained on language pairs with dissimilar labeling schemes, which used multilingual word vectors to process language-mixed data in a shared representation space. We found that joint crosslingual training can automatically learn to bridge the gap between languages and annotation schemas. Then, Chapter 3 applied the crosslingual joint training approach to language models to produce multilingual contextualized word representations. Our experiments showed that these representations could be used to improve results on structured prediction tasks such as semantic role labeling, named entity recognition, and Universal Dependencies parsing even in comparatively high-resource settings (Sections 3.3-3.6) but more significantly in low-resource ones (Sections 3.7-3.8), outperforming other approaches. This demonstrated the relevance of polyglot training to low-resource NLP.

In Part II, we discussed models that addressed the low-resource problem through compositional word representations, sharing statistical strength between rare and common words. GroC (Chapter 4) used a word's character sequence, related words, and definition to generate a representation that also informed the representations of related words. This parameterization led to improved learning for rare words and improved language modeling on small datasets overall, including for domain adaptation, indicating that compositional representations could be useful for modeling low-resource languages. Finally, Chapter 5 extended GroC to a multilingual setting, showing that compositional embeddings could also benefit from polyglot training. Models using this combined strategy achieved the best language modeling results of any model type in our comparison, even outperforming baseline models with twice as much data. We also showed that multilingual compositional embeddings could be aligned crosslingually better than traditional lookup

embeddings, suggesting that vocabulary-independent models could have benefits for applications outside of language modeling.

6.2 Future work

Refining language similarity This work, as well as other contemporary work (Pires et al., 2019; Singh et al., 2019; Lauscher et al., 2020, inter alia) implies that transfer via polyglot training is more effective between related languages (see Chapters 3 and 5). Further examination of whether the same aspects of language similarity are equally important for all tasks could improve the selection of source languages for polyglot models. Alternatively, narrowing down the relevant aspects in which languages must be similar for good crosslingual transfer may provide insight in how to bridge the gap between more distantly related languages.

Language modeling with even less text Multilingual GroC (Chapter 5) enabled the creation of multilingual, vocabulary-independent word embedding spaces. A key feature of vocabulary independence is that representations can be created for words unseen during training of the compositional embedding function. While our experiments with crosslingual lexicons in Chapter 5 did not show large improvements over simple polyglot training without crosslingual supervision, future work could lean more extensively on such supervision to remedy training data imbalance. For example, target language words present in a crosslingual lexicon could be used to improve the target language embedding space via crosslingual alignment, similar to (Faruqui et al., 2015), *whether or not* they are present in the target language corpus. In the limit, a target language embedding space could theoretically be constructed solely from a crosslingual lexicon and source language embeddings, yet extend to cover additional target language words not present in the lexicon. Further research along these lines could be important for handling languages with linguistic documentation yet small amounts of digitized text.

Bibliography

Judit Ács. 2019. Exploring BERT's vocabulary.

Wasi Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard Hovy, Kai-Wei Chang, and Nanyun Peng. 2019. On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In *Proceedings of NAACL-HLT*.

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. 2019. Character-level language modeling with deeper self-attention. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, Honolulu, HI, USA.

Hanan Aldarmaki and Mona Diab. 2019. Context-aware cross-lingual mapping. In *Proceedings of NAACL-HLT*.

Waleed Ammar. 2016. *Towards a Universal Analyzer of Natural Languages*. Ph.D. thesis, Carnegie Mellon University.

Waleed Ammar, Phoebe Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A Smith. 2016a. Many languages, one parser. *TACL*.

Waleed Ammar, Phoebe Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith. 2016b. Massively multilingual word embeddings. *arXiv preprint arXiv:1602.01925*.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of ACL*.

Duygu Ataman, Wilker Aziz, and Alexandra Birch. 2020. A latent morphology model for open-vocabulary neural machine translation. In *Proceedings of ICLR*.

Alexei Baevski and Michael Auli. 2019. Adaptive input representations for neural language modeling. In *Proceedings of ICLR*, New Orleans, LA, USA.

Dzmitry Bahdanau, Tom Bosc, Stanislaw Jastrzebski, Edward Grefenstette, Pascal Vincent, and Yoshua Bengio. 2017. Learning to compute word embeddings on the fly. *CoRR*, abs/1706.00286.

L. R. Bahl, F. Jelinek, and R. L. Mercer. 1983. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(2):179–190.

Kenneth R Beesley and Lauri Karttunen. 2003. Finite-state morphology: Xerox tools and techniques. *CSLI, Stanford*.

Emily M. Bender, Dan Flickinger, and Stephan Oepen. 2002. The grammar matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *COLING-02: Grammar Engineering and Evaluation*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *JMLR*, 3:1137–1155.

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

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 Workshop on Statistical Machine Translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58.

Francis Bond and Ryan Foster. 2013. Linking and extending an open multilingual Wordnet. In *Proceedings of ACL*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*.

Jose Camacho-Collados, Mohammad Taher Pilehvar, Nigel Collier, and Roberto Navigli. 2017. Semeval-2017 task 2: Multilingual and cross-lingual semantic word similarity. In *Proceedings of SemEval-2017*.

Ethan C. Chau, Lucy H. Lin, and Noah A. Smith. 2020. Parsing with multilingual BERT, a small corpus, and a small treebank. In *Findings of the ACL: EMNLP*.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Colin Cherry, George Foster, Ankur Bapna, Orhan Firat, and Wolfgang Macherey. 2018. Revisiting character-based neural machine translation with capacity and compression.

In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4295–4305, Brussels, Belgium. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Kenneth Church and Robert L Mercer. 1993. Introduction to the special issue on computational linguistics using large corpora. *Computational linguistics*, 19(1):1–24.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proceedings of ICLR*.

Shay B Cohen, Dipanjan Das, and Noah A. Smith. 2011. Unsupervised structure prediction with non-parallel multilingual guidance. In *Proceedings of EMNLP*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *Proceedings of ICLR*. Https://openreview.net/forum?id=H196sainb.

Silviu Cucerzan and David Yarowsky. 2000. Language independent, minimally supervised induction of lexical probabilities. In *Proceedings of ACL*, pages 270–277, Hong Kong. Association for Computational Linguistics.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of ACL*.

Hal Daume III. 2007. Frustratingly easy domain adaptation. In *Proceedings of ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Mona Diab and Philip Resnik. 2002. An unsupervised method for word sense tagging using parallel corpora. In *Proceedings of ACL*, pages 255–262, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Timothy Dozat and Christopher Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of ICLR*.

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*.

Philipp Dufter and Hinrich Schütze. 2020. Identifying elements essential for BERT's multilinguality. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4423–4437, Online. Association for Computational Linguistics.

Kevin Duh, Akinori Fujino, and Masaaki Nagata. 2011. Is machine translation ripe for cross-lingual sentiment classification? In *ACL*.

Susan T Dumais, Todd A Letsche, Michael L Littman, and Thomas K Landauer. 1997. Automatic cross-language retrieval using latent semantic indexing. In *AAAI spring symposium on cross-language text and speech retrieval*, volume 15, page 21.

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip Cohen, and Mark Johnson. 2017. Multilingual semantic parsing and code-switching. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 379–389, Vancouver, Canada. Association for Computational Linguistics.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of ACL-IJCNLP*.

Greg Durrett, Adam Pauls, and Dan Klein. 2012. Syntactic transfer using a bilingual lexicon. In *EMNLP*.

David M. Eberhard, Gary F. Simons, and Charles D. Fennig, editors. 2021. *Ethnologue: Languages of the World*, twenty-fourth edition. SIL International, Dallas, TX, USA.

Sergey Edunov, Alexei Baevski, and Michael Auli. 2019. Pre-trained language model representations for language generation. In *Proceedings of NAACL*.

Agnieszka Falenska and Özlem Çetinoğlu. 2017. Lexicalized vs. delexicalized parsing in low-resource scenarios. In *Proceedings of IWPT*.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of NAACL-HLT*.

Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In *Proceedings of EACL*.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Daniela Gerz, Ivan Vulić, Edoardo Ponti, Jason Naradowsky, Roi Reichart, and Anna Korhonen. 2018a. Language modeling for morphologically rich languages: Character-aware modeling for word-level prediction. *Transactions of the Association for Computational Linguistics*, 6:451–465.

Daniela Gerz, Ivan Vulić, Edoardo Maria Ponti, Roi Reichart, and Anna Korhonen. 2018b. On the relation between linguistic typology and (limitations of) multilingual language modeling. In *Proceedings of EMNLP*.

Filip Ginter, Jan Hajič, Juhani Luotolahti, Milan Straka, and Daniel Zeman. 2017. CoNLL 2017 shared task - automatically annotated raw texts and word embeddings. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 languages. In *Proceedings of LREC*.

Joshua T. Goodman. 2001. A bit of progress in language modeling. *Computer Speech and Language*, 15(4):403–434.

Edouard Grave, Moustapha Cisse, and Armand Joulin. 2017a. Unbounded cache model for online language modeling with open vocabulary. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6044–6054, USA. Curran Associates Inc.

Édouard Grave, Armand Joulin, Moustapha Cissé, David Grangier, and Hervé Jégou. 2017b. Efficient softmax approximation for GPUs. In *Proceedings of ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1302–1310, International Convention Centre, Sydney, Australia. PMLR.

Edouard Grave, Armand Joulin, and Nicolas Usunier. 2017c. Improving neural language models with a continuous cache. In *Proceedings of ICLR*, Toulon, France.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Kristina Gulordava, Laura Aina, and Gemma Boleda. 2018. How to represent a word and predict it, too: Improving tied architectures for language modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2936–2941, Brussels, Belgium. Association for Computational Linguistics.

Jiang Guo, Wanxiang Che, Haifeng Wang, Ting Liu, and Jun Xu. 2016a. A unified architecture for semantic role labeling and relation classification. In *Proceedings of COLING*.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2015. Cross-lingual dependency parsing based on distributed representations. In *Proceedings of ACL-IJCNLP*.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2016b. A representation learning framework for multi-source transfer parsing. In *Proceedings of AAAI*.

Mandy Guo, Zihang Dai, Denny Vrandecic, and Rami Al-Rfou. 2020. Wiki-40b: Multilingual language model dataset. In *LREC 2020*.

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 ACL*, pages 8342–8360, Online. Association for Computational Linguistics.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, et al. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL*.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of ACL*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of ICML*.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara I. Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*.

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2017. Tying word vectors and word classifiers: A loss framework for language modeling. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France.

Frederick Jelinek. 1997. *Statistical Methods for Speech Recognition*. MIT Press.

Anders Johannsen, Héctor Martínez Alonso, and Anders Søgaard. 2015. Any-language frame-semantic parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2062–2066, Lisbon, Portugal. Association for Computational Linguistics.

Aravind K Joshi. 1991. Natural language processing. *Science*, 253(5025):1242–1249.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

Karthikeyan K, Zihan Wang, Stephen Mayhew, and Dan Roth. 2020. Cross-lingual ability of multilingual bert: An empirical study. In *Proceedings of ICLR*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *Proceedings of ICLR*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of AAAI*.

Diederik P. Kingma and Jimmy Lei Ba. 2015. ADAM: A method for stochastic optimization. In *Proceedings of ICLR*.

Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. 2012. Inducing crosslingual distributed representations of words. *Proceedings of COLING*.

R. Kneser and H. Ney. 1995. Improved backing-off for m-gram language modeling. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184 vol.1.

Ben Krause, Emmanuel Kahembwe, Iain Murray, and Steve Renals. 2018. Dynamic evaluation of neural sequence models. In *Proceedings of ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 2766–2775, Stockholmsmässan, Stockholm Sweden. PMLR.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of ACL*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Sachin Kumar and Yulia Tsvetkov. 2018. Von mises-fisher loss for training sequence to sequence models with continuous outputs. In *International Conference on Learning Representations*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of EMNLP*.

Miryam de Lhoneux, Johannes Bjerva, Isabelle Augenstein, and Anders SÃ¸gaard. 2018. Parameter sharing between dependency parsers for related languages. In *Proceedings of EMNLP*.

Miryam de Lhoneux, Yan Shao, Ali Basirat, Eliyahu Kiperwasser, Sara Stymne, Yoav Goldberg, and Joakim Nivre. 2017. From raw text to universal dependencies - look, no tags! In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Gurvan L'Hostis, David Grangier, and Michael Auli. 2016. Vocabulary selection strategies for neural machine translation. *arXiv preprint arXiv:1610.00072*.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, et al. 2020. Xglue: A new benchmark dataset for cross-lingual pre-training, understanding and generation. In *Proceedings of EMNLP*.

KyungTae Lim, Cheoneum Park, Changki Lee, and Thierry Poibeau. 2018. SEx BiST: A multi-source trainable parser with deep contextualized lexical representations. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

KyungTae Lim and Thierry Poibeau. 2017. A system for multilingual dependency parsing based on bidirectional LSTM feature representations. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramón Fermandez, Silvio Amir, Luís Marujo, and Tiago Luís. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of EMNLP*, pages 1520–1530, Lisbon, Portugal. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of NAACL-HLT*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of ACL, System Demonstrations*, pages 55–60.

Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. *arXiv preprint arXiv:1701.02593*.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of EMNLP*, Copenhagen, Denmark.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The Penn treebank. *Computational Linguistics*.

Austin Matthews, Graham Neubig, and Chris Dyer. 2018. Using morphological knowledge in open-vocabulary neural language models. 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 1435–1445, New Orleans, Louisiana. Association for Computational Linguistics.

John McCarthy, Marvin L Minsky, Nathaniel Rochester, and Claude E Shannon. 1955. A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of ACL*.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of EMNLP*.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. Regularizing and optimizing LSTM language models. In *International Conference on Learning Representations*, Vancouver, Canada.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France.

Sabrina J. Mielke and Jason Eisner. 2018. Spell once, summon anywhere: A two-level open-vocabulary language model. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, volume abs/1804.08205, Honolulu, HI, USA.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048. ISCA.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*.

Celia M Millward and Mary Hayes. 2012. *A Biography of the English Language (Third Edition)*. Wadsworth Cengage Learning.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *International workshop on artificial intelligence and statistics*, pages 246–252. Proceedings of MLR.

Raihan Muhamedowa. 2015. *Kazakh: A comprehensive grammar*. Routledge.

Phoebe Mulcaire, Jungo Kasai, and Noah A Smith. 2019a. Low-resource parsing with crosslingual contextualized representations. In *Proceedings of CoNLL*.

Phoebe Mulcaire, Jungo Kasai, and Noah A. Smith. 2019b. Polyglot contextual representations improve crosslingual transfer. In *Proceedings of ACL*.

Phoebe Mulcaire, Swabha Swayamdipta, and Noah A Smith. 2018. Polyglot semantic role labeling. In *Proceedings of ACL*, volume 2, pages 667–672.

Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective sharing for multilingual dependency parsing. In *Proceedings of ACL*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan T. McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of LREC*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of LREC*, pages 4034–4043, Marseille, France. European Language Resources Association.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2020. What the [mask]? making sense of language-specific bert models. *arXiv preprint arXiv:2003.02912*.

Robert Östling and Jörg Tiedemann. 2017. Continuous multilinguality with language vectors. In *Proceedings of EACL*.

Sebastian Padó and Mirella Lapata. 2005. Cross-linguistic projection of role-semantic information. In *Proceedings of HLT-EMNLP*.

Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *JAIR*, 36:307–340.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Nikolaos Pappas and James Henderson. 2019. Deep residual output layers for neural language generation. In *Proceedings of ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 5000–5011, Long Beach, California, USA. PMLR.

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. Association for Computational Linguistics.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018. Learning joint semantic parsers from disjoint data. In *Proceedings of NAACL-HLT*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of ACL*, pages 1756–1765.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations.

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 43–54, Hong Kong, China. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of ACL*.

Edoardo Maria Ponti, Ivan Vulić, Ryan Cotterell, Roi Reichart, and Anna Korhonen. 2019. Towards zero-shot language modeling. 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 2893–2903, Hong Kong, China. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of CoNLL*, pages 143–152.

Ofir Press, Noah A. Smith, and Omer Levy. 2020. Improving transformer models by reordering their sublayers. In *Proceedings of ACL*, pages 2996–3005, Online. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of EACL*.

Peter Prettenhofer and Benno Stein. 2011. Cross-lingual adaptation using structural correspondence learning. *TIST*.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170.

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 ACL*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Anton Ragni, Edgar Dakin, Xie Chen, Mark J. F. Gales, and Kate Knill. 2016. Multi-language neural network language models. In *Proceedings of INTERSPEECH*, volume 08-12-September-2016, pages 3042–3046.

Mohammad Sadegh Rasooli and Michael Collins. 2017. Cross-lingual syntactic transfer with limited resources. *TACL*, 5.

Philip Resnik, Mari Broman Olsen, and Mona Diab. 1999. The bible as a parallel corpus: Annotating the 'book of 2000 tongues'. *Computers and the Humanities*, 33(1):129–153.

Kyle Richardson, Jonathan Berant, and Jonas Kuhn. 2018. Polyglot semantic parsing in APIs. 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 720–730, New Orleans, Louisiana. Association for Computational Linguistics.

Rudolf Rosa and David Mareček. 2018. CUNI x-ling: Parsing under-resourced languages in CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of ACL*, pages 1192–1202, Berlin, Germany. Association for Computational Linguistics.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2017. A survey of cross-lingual word embedding models. *arXiv preprint arXiv:1706.04902*.

Timo Schick and Hinrich Schütze. 2020. BERTRAM: Improved word embeddings have big impact on contextualized model performance. In *Proceedings of ACL*, pages 3996–4007, Online. Association for Computational Linguistics.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of NAACL-HLT*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of ACL*, pages 1715–1725, Berlin, Germany.

Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *Proceedings ICLR*.

Jasdeep Singh, Bryan McCann, Richard Socher, and Caiming Xiong. 2019. BERT is not an interlingua and the bias of tokenization. In *Proceedings of DeepLo 2019*.

Aaron Smith, Bernd Bohnet, Miryam de Lhoneux, Joakim Nivre, Yan Shao, and Sara Stymne. 2018. 82 treebanks, 34 models: Universal dependency parsing with multi-treebank models. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *NIPS*.

Milan Straka, Jan Hajic, and Jana Straková. 2016. UDPipe: Trainable pipeline for processing conll-u files performing tokenization, morphological analysis, pos tagging and parsing. In *Proceedings of LREC*, pages 4290–4297.

Sainbayar Sukhbaatar, Edouard Grave, Piotr Bojanowski, and Armand Joulin. 2019. Adaptive attention span in transformers. In *Proceedings of ACL*, pages 331–335, Florence, Italy. Association for Computational Linguistics.

Swabha Swayamdipta, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Greedy, joint syntactic-semantic parsing with stack LSTMs. In *Proceedings of CoNLL*.

Swabha Swayamdipta, Sam Thomson, Kenton Lee, Luke Zettlemoyer, Chris Dyer, and Noah A. Smith. 2018. Syntactic scaffolds for semantic structures. In *Proceedings of EMNLP*.

Oscar Täckström, Ryan McDonald, and Joakim Nivre. 2013. Target language adaptation of discriminative transfer parsers. In *Proceedings of NAACL-HLT*.

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of NAACL-HLT*.

Mariona Taulé, M. Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel annotated corpora for Catalan and Spanish. In *Proceedings of LREC*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of ACL*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Yulia Tsvetkov, Sunayana Sitaram, Manaal Faruqui, Guillaume Lample, Patrick Littell, David Mortensen, Alan W Black, Lori Levin, and Chris Dyer. 2016. Polyglot neural language models: A case study in cross-lingual phonetic representation learning. In *Proceedings of NAACL-HLT*, pages 1357–1366.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. 2016. Cross-lingual models of word embeddings: An empirical comparison. In *Proceedings of ACL*.

Clara Vania, Xingxing Zhang, and Adam Lopez. 2017. UParse: the Edinburgh system for the CoNLL 2017 UD shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

David Vilares, Carlos Gómez-Rodríguez, and Miguel A. Alonso. 2016. One model, two languages: training bilingual parsers with harmonized treebanks. In *Proceedings of ACL*.

Hui Wan, Tahira Naseem, Young-Suk Lee, Vittorio Castelli, and Miguel Ballesteros. 2018. IBM research at the CoNLL 2018 shared task on multilingual parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Dingquan Wang and Jason Eisner. 2016. The galactic dependencies treebanks: Getting more data by synthesizing new languages. *TACL*, 4.

Dingquan Wang and Jason Eisner. 2018a. Surface statistics of an unknown language indicate how to parse it. *TACL*.

Dingquan Wang and Jason Eisner. 2018b. Synthetic data made to order: The case of parsing. In *Proceedings of EMNLP*.

Warren Weaver. 1952. Translation. In *Proceedings of the Conference on Mechanical Translation*, Massachusetts Institute of Technology.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual BERT? In *Proceedings of the 5th Workshop on Representation Learning for NLP*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of EMNLP*, pages 369–379.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of NAACL-HLT*.

David Yarowsky, Grace Ngai, and Richard H Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *HLT*.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of IJCNLP Workshop on NLP for Less Privileged Languages*.

Yuan Zhang and Regina Barzilay. 2015. Hierarchical low-rank tensors for multilingual transfer parsing. In *Proceedings of EMNLP*.

Hai Zhao, Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Multilingual dependency learning: Exploiting rich features for tagging syntactic and semantic dependencies. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 61–66, Boulder, Colorado. Association for Computational Linguistics.

Julian Georg Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. 2017. Recurrent highway networks. In *Proceedings of ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 4189–4198, International Convention Centre, Sydney, Australia. PMLR.

Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*.

Appendix A

POLYGLOT LANGUAGE MODELS (SUPPLEMENTARY)

In this supplementary material, we provide hyperparameters used in our models for easy replication of our results.

A.1 Language Models

Seen in Table A.1 is a list of hyperparameters for our language models. We generally follow Peters et al. (2018) and use their publicly available code for training.[1] For character only models, we halve the LSTM and projection sizes to expedite training and to compensate for the greatly reduced training data—their hyperparameters were tuned on around 30M sentences, while we used less than 3M sentences (60-70M tokens) per language.

A.2 UD Parsing

For UD parsing, we generally follow the hyperparameters provided in AllenNLP (Gardner et al., 2018). See a list of hyperparameters in Table A.3. We use stratified sampling so that each training mini-batch has an equal number of sentences from the source and target languages.

A.3 Semantic Role Labeling

For SRL, we again follow the hyperparameters given in AllenNLP (Table A.2). The one exception is that we used 4 layers of alternating BiLSTMs instead of 8 layers to expedite the training process.

A.4 Named Entity Recognition

We again use the hyperparameter configurations provided in AllenNLP. See Table A.4 for details.

A.4.1 Multilingual Word Vectors

We train our word type representations used for non-contextual baselines with fastText (Bojanowski et al., 2017). We use window size 5 and a minimum count of 5, with 300

[1] https://github.com/allenai/bilm-tf

Character CNNs	
Char embedding size	16
(# Window Size, # Filters)	(1, 32), (2, 32), (3, 68), (4, 128), (5, 256), 6, 512), (7, 1024)
Activation	Relu
Word-level LSTM	
LSTM size	2048
# LSTM layers	2
LSTM projection size	256
Use skip connections	Yes
Inter-layer dropout rate	0.1
Training	
Batch size	128
Unroll steps (Window Size)	20
# Negative samples	64
# Epochs	10
Adagrad (Duchi et al., 2011) lrate	0.2
Adagrad initial accumulator value	1.0

Table A.1: Language Model Hyperparameters.

dimensions.

A.5 Other Low-Resource Simulations

In addition to the 100-sentence low-resource condition described in Section 3.8, we simulated low-resource experiments with 500 and 1000 sentences of target language data, and zero-target-treebank experiments in which the parser was trained with only source language data, but with multilingual representations allowing crosslingual transfer. See Table A.5 for these results. The additional low-resource results confirm our analysis in Section 3.8.2: polyglot training is more effective the less target-language data is available, with a slight advantage for related languages.

A.6 UD Treebanks

Additional statistics about the languages and treebanks used are given in Table A.6.

Input	
Predicate indicator emb size	100
Word-level Alternating BiLSTM	
LSTM size	300
# LSTM layers	4
Recurrent dropout rate	0.1
Use Highway Connection	Yes
Training	
Batch size	80
# Epochs	80
Early stopping	20
Adadelta (Zeiler, 2012) lrate	0.1
Adadelta ρ	0.95
Gradient clipping	1.0

Table A.2: SRL hyperparameters.

Input	
POS embedding size (when used)	100
Input dropout rate	0.3
Word-level BiLSTM	
LSTM size	400
# LSTM layers	3
Recurrent dropout rate	0.3
Inter-layer dropout rate	0.3
Use Highway Connection	Yes
Multilayer Perceptron, Attention	
Arc MLP size	500
Label MLP size	100
# MLP layers	1
Activation	Relu
Training	
Batch size	80
# Epochs	80
Early stopping	50
Adam (Kingma and Ba, 2015) lrate	0.001
Adam β_1	0.9
Adam β_2	0.999

Table A.3: UD Parsing Hyperparameters.

Char-level LSTM	
Char embedding size	25
Input dropout rate	0.5
LSTM size	128
# LSTM layers	1
Word-level BiLSTM	
LSTM size	200
# LSTM layers	3
Inter-layer dropout rate	0.5
Recurrent dropout rate	0.5
Use highway connection	Yes
Multilayer Perceptron	
MLP size	400
Activation	tanh
Training	
Batch size	64
# Epochs	50
Early stopping	25
Adam (Kingma and Ba, 2015) lrate	0.001
Adam β_1	0.9
Adam β_2	0.999
L2 regularization coefficient	0.001

Table A.4: NER hyperparameters.

	$\lvert D_\tau \rvert = 0$		$\lvert D_\tau \rvert = 100$			$\lvert D_\tau \rvert = 500$			$\lvert D_\tau \rvert = 1000$		
target	+eng	+rel.	mono	+eng	+rel.	mono	+eng	+rel.	mono	+eng	+rel.
ARA	10.31	**20.47**	62.50	73.39	**73.43**	76.15	**79.55**	79.16	79.43	81.38	**81.49**
HEB	23.76	**24.89**	64.53	74.86	**75.69**	79.27	82.35	**82.92**	82.59	84.59	**84.70**
HRV	48.69	**67.67**	63.49	79.21	**82.00**	80.80	84.92	**85.89**	84.14	86.27	**86.66**
RUS	38.69	**73.24**	59.51	75.63	**79.29**	77.38	83.16	**84.60**	82.90	85.68	**86.99**
NLD	61.68	**72.90**	57.12	74.90	**77.01**	75.19	**82.42**	81.33	81.41	**84.93**	83.23
DEU	51.18	**68.66**	60.26	72.52	**73.45**	72.94	**77.88**	77.68	76.46	**78.67**	78.57
SPA	55.85	**75.88**	64.97	80.86	**81.55**	79.67	**84.88**	84.63	82.97	86.69	**86.81**
ITA	59.71	**78.12**	69.17	**84.63**	83.51	82.96	**88.96**	87.91	87.03	**90.22**	89.32
CMN	**8.16**	5.34	53.36	**63.63**	61.47	71.94	74.88	**74.98**	77.42	**79.07**	78.96
JPN	4.12	**11.66**	72.37	**80.94**	80.24	86.20	**87.74**	**87.74**	88.74	89.08	**89.32**

Table A.5: LAS for UD parsing with additional simulated low-resource and zero-target-treebank settings.

Lang	Code	WALS Genus	WALS 81A	Size (# sents.)	Treebank	Genre
English	eng	Germanic	SVO		EWT	blog, email, reviews, social
			Simulation Pairs			
Arabic	ara	Semitic	VSO/SVO	5241	PADT	news
Hebrew	heb	Semitic	SVO		HTB	news
Croatian	hrv	Slavic	SVO	6983	SET	news, web, wiki
Russian	rus	Slavic	SVO		SynTagRus	contemporary fiction, popular, science, newspaper, journal articles, online news
Dutch	nld	Germanic	SOV/SVO	12269	Alpino	news
German	deu	Germanic	SOV/SVO		GSD	news, reviews, wiki
Spanish	spa	Romance	SVO	12543	GSD	blog, news, reviews, wiki
Italian	ita	Romance	SVO		ISDT	legal, news, wiki
Chinese	cmn	Chinese	SVO	3997	GSD	wiki
Japanese	jpn	Japanese	SOV		GSD	wiki
			Truly Low Resource and Related Languages			
Hungarian	hun	Ugric	SOV/SVO	910	Szeged	news
Finnish	fin	Finnic	SVO	12217	TDT	news, wiki, blog, legal, fiction, grammar-examples
Vietnamese	vie	Viet-Muong	SVO	1400	VTB	news
Uyghur	uig	Turkic	SOV	1656	UDT	fiction
Kazakh	kaz	Turkic	SOV	31	KTB	wiki, fiction, news
Turkish	tur	Turkic	SOV	3685	IMST	nonfiction, news

Table A.6: List of the languages and their UD treebanks used in our experiments (expanded from Table 3.3).

Appendix B

GROUNDED COMPOSITIONAL OUTPUTS (SUPPLEMENTARY)

We report here the computer infrastructure and experimental details, including hyperparameter bounds, hyperparameter optimal values, training speed, development scores, for all of the experiments in Chapter 4 where applicable. We also provide a comparison with state of the art by taking into account the number of model parameters and guide the reader through the replication effort we did to reproduce the neural cache by Grave et al. (2017a).

B.1 Conventional Language Modeling

For the experiments with a closed vocabulary on `penn`[1] and `wikitext-2`,[2] we used the following computing infrastructure: 5 GeForce RTX 2080 Ti gpu cards. Our codebase is based on Pytorch[3] and is publicly available on Github.[4]

B.1.1 Model Configuration

The prefix network used by all output embedding methods is a vanilla recurrent neural network based on the implementation by Merity et al. (2017)[5] with 2 layers and 1024 LSTM units, regularized with hidden unit dropout of 0.65 along the lines of Grave et al. (2017a). The maximum length of the relational and definitional forms from Wordnet is set to 3 and 10 without search based on our computational budget.[6] The embedding size is set to 300 for `penn` and 256 for `wikitext2`. For optimization we use Adam with a learning rate of 0.001, initial weight uniformly sampled in the range $[-0.05, 0.05]$, and a batch size of 20 for `penn` and `wikitext2`. We clip the norm of the gradient to 0.1 and unroll the network for 35 steps. The learning rate is multiplied by 0.1 if the development loss does not decrease for 4 consecutive epochs and we perform early stopping if there is no improvement for 8 consecutive epochs.

---

[1] `www.fit.vutbr.cz/~imikolov/rnnlm/simple-examples.tgz`

[2] `s3.amazonaws.com/research.metamind.io/wikitext/wikitext-2-v1.zip`

[3] `pytorch.org/get-started`

[4] `github.com/Noahs-ARK/groc`

[5] `github.com/salesforce/awd-lstm-lm`

[6] We expect that a larger budget would generally allow to increase these limits and obtain even better results.

Hyperparameter	abbrev.	range	trials
Output dropout	r	$\{0, 0.1, \ldots, 1.0\}$	10
Linear correction	cor	$\{32, 64, 128\}$	3
Adaptive cutoffs	cut	$\{253, 721, 118, 226, 424, 334\}$	6
Output net depth	k	$\{0, 1, 2, 3, 4\}$	4
Output net activation	act	$\{relu, selu, tanh\}$	3

Table B.1: Hyperparameters, range of values, and, number of trials required to search them. Adaptive cutoffs are read as follows: e.g. for 253 the cutoff array contains $\lceil 0.2 * n, 0.5 * n, 0.3 * n \rceil, n = |\mathcal{V}|$ words per bin.

B.1.2 Hyperparameter Optimization

For all methods, the hyperparameter selection of output embedding dropout rate (r), output network depth (k) and activation (act), linear "correction", and adaptive frequency cutoffs was conducted by grid search over specific range of values given in Table B.1 on development data. Note that not all the hyperparameters apply to all methods, as can be seen in Table B.2 where we report the optimal hyperparameter values for each of the methods. For all the baselines we performed exhaustive grid search on both datasets, but for our method we performed grid search only on `penn` and searched manually on `wikitext-2` by selecting values of hyperparameters that were ranked high based on the grid search on `penn` to avoid the increased cost that comes with training our method (see speed comparison in Appendix B.1.4). The total number of trials for all methods including our ablations were 204 and 67 respectively for `penn` and `wikitext-2` respectively. Note that the reduced number of trials is due to not performing exhaustive search for our method and its ablations as explained above. The number of trials per method can be derived by multiplying the non-zero columns per row with the number of trials required for each column.

B.1.3 Development Scores

Table B.3 displays the development scores and number of parameters along with the test perplexities for our model and all the baseline output embedding methods for our main experiment. The development scores for the models of the ablation study and for the base models of the coverage experiment have already been given in Table 4.3 in the main paper (Section 4.5.3). Overall, we can observe that in most cases the ranking based on the development scores is indicative of the ranking of the methods according to the test scores.

	penn					wikitext2				
Method	r	cor	cut	k	act	r	cor	cut	k	act
Lookup table	0.1	–	–	–	–	0.2	–	–	–	–
Convolutional	0.1	128	–	–	–	0.1	182	–	–	–
Tied	0.0	–	–	–	–	0.0	–	–	–	–
Bilinear	0.5	–	–	–	–	0.4	–	–	–	–
Deep residual	0.5	–	–	4	selu	0.6	–	–	1	selu
Adaptive	0.3	–	2k7k	–	–	0.2	–	6k21k	–	–
GroC (ours)	0.2	–	–	0	–	0.2	–	–	1	relu
− relations	0.3	–	–	0	–	0.3	–	–	1	selu
− definitions	0.2	–	–	0	–	0.3	–	–	2	relu
− both	0.3	–	–	0	–	0.3	–	–	1	selu

Table B.2: Best hyperparameter values per method.

	penn			wikitext2		
Method	$\lvert\Theta\rvert$	dev.	test	$\lvert\Theta\rvert$	dev.	test
Lookup table	13M	93.5	90.8	23M	113.8	108.3
Convolutional	13M	104.0	101.6	23M	121.2	116.6
Tied	10M	88.6	86.2	15M	101.0	97.3
Bilinear	10M	87.0	83.7	15M	101.3	95.9
Deep residual	10M	84.0	80.5	15M	100.1	94.7
Adaptive	8M	84.0	79.3	9M	95.8	90.7
GroC (ours)	9M	72.5	69.5	9M	87.0	82.5

Table B.3: Development and test scores on conventional language modeling benchmarks with closed vocabulary. $\lvert\Theta\rvert$ denotes the total number of model parameters.

B.1.4 Training Speed

Table B.4 displays the average training speed per epoch in seconds for each of the methods. This experiment was run on a single, dedicated[7] GeForce RTX 2080 Ti. As we mentioned in Section 4.4.1, even though our model has vocabulary-size independent parameterization it is not independent of the computation that is required to encode the vocabulary. This has a negative impact on the training speed of GroC, making it a bit slower than the Convolutional method, namely $1.3\times$ slower.

To mitigate this problem we recommend training GroC with sparse updates for the output embedding parameters as described in the main paper (Section 4.4.2). Concretely, at each training iteration with probability p we make a full update and keep the output embedding frozen otherwise. The rest of the network is trained with full updates as before. We can observe that this optimization strategy makes GroC nearly as efficient as the baselines with $p = 0.1$ or $p = 0.3$. In particular, it becomes even faster than the convolutional baseline by $2.1\times$. Furthermore, our best model with $p = 0.3$ which is much faster reaches 75.3 perplexity on `penn` without additional hyperparameter optimization which is still -4 points lower than the second best, adaptive output embedding; tuning the model from scratch should likely lead to even better results. This is quite encouraging because it means that the benefits of our model need not come with a large computational cost. In future work, the training speed could be optimized even further by devising specialized efficient training methods for compositional outputs.

B.1.5 Comparison with State-of-the-Art Models

Table B.5 displays several state-of-the-art models which have number of parameters ranging from 9M to 20M on Penn Treebank. We can observe that our model which has only 9.7M parameters achieves better performance than all the models that have lower than or equal to 21M parameters and even the model by Inan et al. (2017) which has 24M parameters. Note that our model has lower perplexity than the pointer sentinel mixture model by Merity et al. (2017) and the neural cache model by Grave et al. (2017a) while having 11M less parameters than them.

Moreover, it is very close to the other models which have around 23-25M parameters without being highly regularized (weight dropout, input dropout) or having advanced optimization strategies (SGD + ASGD, finetuning) like AWD-LSTM (Merity et al., 2017). Training larger models and investigating the potential of competing with even higher capacity models is an interesting direction which we hope will be explored in future studies.

---

[7]By dedicated GPU card here we mean that no other processes were using the GPU card when we performed the experiments for each of the methods.

Method	`penn`	`wikitext-2`
Lookup table	19.5	59.5
Convolutional	201.2	1301.9
Tied	18.6	53.6
Bilinear	35.0	120.1
Deep residual	61.2	114.5
Adaptive	27.2	77.6
GroC (ours)	259.8	1813.5
– 10% updates	236.3	1627.7
– 30% updates	173.5	1262.9
– 50% updates	131.5	936.4
– 70% updates	95.2	669.0
– 90% updates	46.0	299.0

Table B.4: Training speed for each method. We report the average time in seconds to complete one epoch.

Model	$\lvert\Theta\rvert$	test
Mikolov and Zweig (2012) – RNN-LDA	9M‡	92.0
Zaremba et al. (2014) – LSTM	20M	82.7
Gal and Ghahramani (2016) – Var. LSTM	20M	78.6
Kim et al. (2016) – CharCNN	19M	78.9
Merity et al. (2017) – Pointer Sentinel-LSTM	21M	70.9
Grave et al. (2017c) – LSTM + cont. cache	-	72.1
Inan et al. (2017) – Tied Variational LSTM	24M	73.2
Zilly et al. (2017) – Variational RHN	23M	65.4
Zoph and Le (2016) – NAS Cell	25M	64.0
Merity et al. (2018) – AWD-LSTM	24M	58.8
Ours – LSTM	10M	86.2
Ours – LSTM + GroC (sur,rel,def)	9.7M	69.5

Table B.5: Comparison with state-of-the-art models of comparable size to that of Grave et al. (2017a) and Merity et al. (2017) on the `penn` dataset.

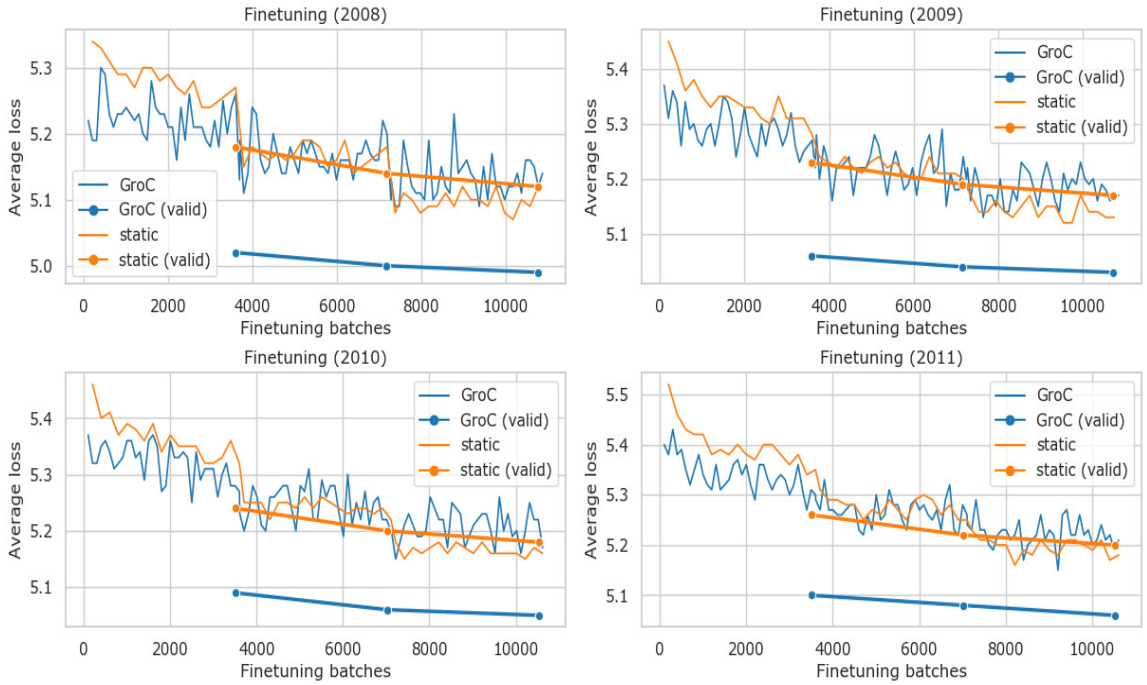

Figure B.1: Training and validation loss for GroC and the tied model during finetuning on near domains.

Model	2007 →	near domains 2008	2009	2010	2011	far domains Web	Wiki
Tied + finetuning	–	167.44	175.95	177.46	180.63	144.13	232.06
Grounded + finetuning	–	146.84	152.29	155.27	158.21	212.99	188.25

Table B.6: Validation perplexity for finetuned models on cross-domain language modeling.

B.2 Cross-Domain Language Modeling

For the experiment in cross-domain language modeling, we used the following computing infrastructure: 2 GeForce RTX 2080 Ti and 2 TITAN RTX GPUs to train and finetune our GroC models, and 2 Tesla P100 GPUs to train and finetune the baselines and to perform hyperparameter search.

B.2.1 Finetuning Dynamics

Figures B.1 and B.2 show the loss on the training and validation data for the target domain during finetuning. GroC generalizes better from the training to the validation data than the tied model, consistently having lower validation loss. The training loss for GroC consistently starts out lower than that the of the tied model, showing that it has less difficulty adapting to the new data, and ends up higher, indicating greater regularization vs the tied model.

The `web` dataset is a clear outlier, in which the tied model improves much more dramatically than in any other domain. The difference in validation performance here is reflected in the test perplexity (Table 4.6) but does not have a clear explanation.

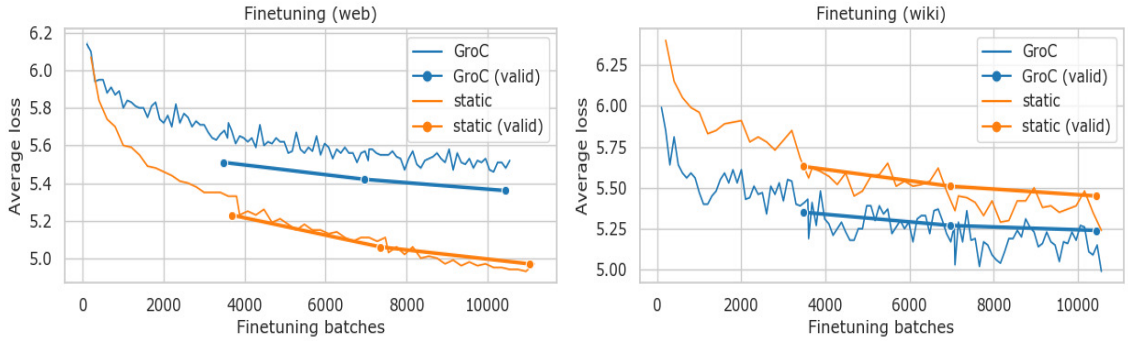

Figure B.2: Training and validation loss for GroC and the tied model during finetuning, on far domains.

B.2.2 Data

As described in Section 4.6, the choice of data and preprocessing used for the cross-domain experiments are based on Grave et al. (2017a). News Crawl and Common Crawl can be downloaded from the WMT 2014 website.[8] WikiText-103 was downloaded from Salesforce website[9]. For the News Crawl datasets, the first 2M tokens of the English data for each year were used as the train set, the next 2M tokens as the validation set, and the next 10M tokens as the test set. The same procedure was used for `web` (Common Crawl), for which we used the English portion of the English-German aligned data. While Grave et al. (2017a) describes the Common Crawl data as shuffled at the sentence level, we found that most sentences seemed closely related to adjacent sentences, so after creating train/valid/test splits for this dataset we re-shuffled each file. WikiText-103 comes divided into train/valid/test splits, so we used the first 2M/2M/10M tokens of each split respectively for our dataset. All data was then tokenized using the Europarl tokenizer[10] and lowercased.

Our data preprocessing can be replicated with the script `create-data.sh`, available with the code for GroC.[11]

B.2.3 Finetuning Validation Results

Because no target-domain training is required for most of our cross-domain experiments, validation scores were not computed for most model-domain combinations; however, we report the validation perplexity for the finetuned models in Table B.6, to aid in replication.

B.2.4 Hyperparameter Selection

Cache hyperparameters were selected via grid search, with θ, the flattening hyperparameter described in Grave et al. (2017c), ranging over 5 values from 0 to 1, and λ ranging over 5 values from 0.833 to 0.966 (bounds which were selected based on the optimal hyperparameter ranges in (Grave et al., 2017c)). Perplexity of a model trained on `2007` and evaluated on the `2008` validation set was the metric used to select the optimal hyperparameters: $\lambda = 0.966$ for unigram and neural cache and $\theta = 0.5$ for neural cache. Because the cache is only used during evaluation, this hyperparameter search was quite efficient to carry out using the tied model, requiring no additional training, only 25 evaluation runs on the validation set. This hyperparameter search is illustrated in Figure B.3.

We then used the same hyperparameters for all cache models. This provides a slight advantage to the tied model, as the optimal hyperparameters for GroC might be different

[8] `www.statmt.org/wmt14/translation-task.html`

[9] `blog.einstein.ai/the-wikitext-long-term-dependency-language-modeling-dataset/`

[10] `statmt.org/europarl/v7/tools.tgz`

[11] `github.com/<anon>/groc`

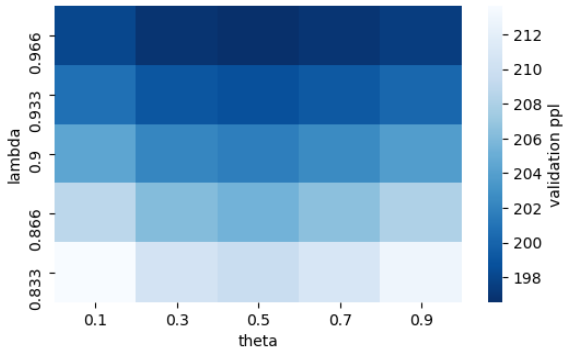

Figure B.3: Validation accuracy for various hyperparameter settings on the `2008` validation set.

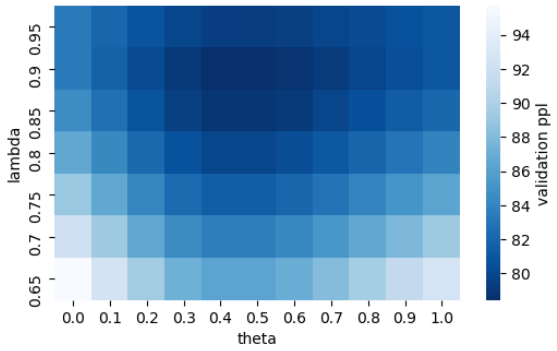

Figure B.4: Validation accuracy for various hyperparameter settings on the `penn` validation set.

from those selected with the tied model. A cache size of 5,000 was used during hyperparameter tuning, but at test time we used 10,000 for all experiments based on its use in Grave et al. (2017a). Figure B.4 shows a separate hyperparameter search performed over the `penn` validation set to confirm the accuracy of our neural cache reimplementation. Compare to Figure 2a in Grave et al. (2017c); note their λ is 1 minus ours.

For GroC, we also selected a downweighting hyperparameter dw, based on validation performance on the `wiki` dataset only. We searched over 5 values (0.1, 0.3, 0.5, 0.7, and 0.9) using GroC with the neural cache, and selected $dw = 0.1$ as the best value with a validation ppl of 154.01.

Appendix C

POLYGLOT COMPOSITIONAL OUTPUT EMBEDDINGS (SUPPLEMENTARY)

C.1 Model Configuration

Our language model configuration is closely based on the one used in Chapter 4. The prefix network used for all models is a recurrent neural network based on the implementation by Merity et al. (2017)[1] with 2 layers and 1024 LSTM units, regularized with hidden unit dropout of 0.65 along the lines of Grave et al. (2017a). We use an embedding size of 256, and sample initial weights uniformly in the range [-0.05,0.05]. Due to the computational cost of training GroC models, we did not perform an independent hyperparameter search on the `wiki40b` datasets used in Chapter 5. We use the hyperparameter values that were found to be optimal for the `penn` dataset in Chapter 4, with no residual network (i.e. depth 0), and an output dropout of 0.2.

For training with a language modeling objective, we use an initial learning rate of 0.001, which is divided by 10 if the development loss does not decrease for 4 consecutive epochs, and perform early stopping if the development loss does not decrease for 8 consecutive epochs.

C.2 Hyperparameter Optimization: ELMo Initialization

For finetuning GroC models initialized from an ELMo-like model (Section 5.5), we performed a hyperparameter search over initial learning rates for the English-French language pair. We searched over 10 values (0.01, $[5, 2, 1] \times [0.001, 0.0001, 0.00001]$) and found that a learning rate of 0.001 performs best (the same as we use for training a model from random initialization) despite the significantly shorter training time required for good performance.

[1] `github.com/salesforce/awd-lstm-lm`