On Understanding Character-level Models for Representing Morphology

Clara Vania

Doctor of Philosophy
Institute for Language, Cognition and Computation
School of Informatics
University of Edinburgh

2020

Abstract

Morphology is the study of how words are composed of smaller units of meaning (*morphemes*). It allows humans to create, memorize, and understand words in their language. To process and understand human languages, we expect our computational models to also learn morphology. Recent advances in neural network models provide us with models that compose word representations from smaller units like word segments, character n-grams, or characters. These so-called subword unit models do not explicitly model morphology yet they achieve impressive performance across many multilingual NLP tasks, especially on languages with complex morphological processes. This thesis aims to shed light on the following questions: (1) What do subword unit models learn about morphology? (2) Do we still need prior knowledge about morphology? (3) How do subword unit models interact with morphological typology? First, we systematically compare various subword unit models and study their performance across language typologies. We show that models based on characters are particularly effective because they learn orthographic regularities which are consistent with morphology. To understand which aspects of morphology are not captured by these models, we compare them with an oracle with access to explicit morphological analysis. We show that in the case of dependency parsing, character-level models are still poor in representing words with ambiguous analyses. We then demonstrate how explicit modeling of morphology is helpful in such cases. Finally, we study how character-level models perform in low resource, cross-lingual NLP scenarios, whether they can facilitate cross-linguistic transfer of morphology across related languages. While we show that cross-lingual character-level models can improve low-resource NLP performance, our analysis suggests that it is mostly because of the structural similarities between languages and we do not yet find any strong evidence of cross-linguistic transfer of morphology. This thesis presents a careful, in-depth study and analyses of character-level models and their relation to morphology, providing insights and future research directions on building morphologically-aware computational NLP models.

Lay Summary

The advancement of technology has made a tremendous impact in our daily lives. Every day people interact with technology to fulfil their information needs, and we expect that one day we can naturally communicate with computer as we communicate with humans. To do so, we first need to build a system that can understand human languages. But human languages are remarkably complex and diverse, which make processing and understanding languages challenging. For example, the same information can be expressed using one word in Turkish but several words, or even one whole sentence in English. In this thesis, we focus on one particular aspect of language called *morphology*, which is the study of how words are composed of smaller units of meaning. We study how words are represented in the current natural language processing systems. We show that models that represent word from characters are effective for many languages, but we also demonstrate that prior knowledge of morphology can further improve the performance of these models.

Acknowledgements

First of all, it is hard to believe that I am at the end of my PhD journey. There are so many people that I want to thank for their support throughout my study.

I am deeply grateful to my supervisor Adam Lopez, for his incredible support, patience, and guidance during my PhD. I would like to thank you for introducing me to the NLP research which I knew nothing about at the beginning of my PhD and for helping me grow as an independent researcher. Thank you for always encouraging me to overcome my fear of maths, for your great patience with my consistent confusions about neural networks, languages, or other random things. It was a really great pleasure to work with you and I am sure I will miss our weekly and group meetings.

I would like to thank my PhD committee, Sharon Goldwater and Mirella Lapata for their feedback and guidance which helped improve the content of my PhD work. I would also like to thank all my collaborators for your contribution to the thesis. Xingxing Zhang, for introducing me to neural dependency parsing and for general advice about neural network training tricks. Andreas Grivas, who was a master student back in 2017, for our collaborations for the dependency parsing work. I would like to thank Yova Kementchedjhieva and Anders Søgaard for inviting me to the University of Copenhagen and for brainstorming about research ideas that shaped my work on low-resource dependency parsing. Finally, I would like to thank my examiners, Frank Keller and Reut Tsarfaty for taking their valuable time for assessing this thesis. Their constructive feedback have helped strengthen and shaping up this thesis. This thesis has greatly benefited from the Wikipedia datasets and the Universal Dependencies treebanks and I would like to thank all of them who have made these resources publicly available, especially for many languages.

It was a great experience to have been studying at the University of Edinburgh. I would like to thank CDT in Data Science, for accepting me as a student and gave great support during my studies. I would like to thank the CDT management: Chris Williams, Charles Sutton, Amos Storkey, Adam Lopez, the administrative staffs Sally Galloway and Brittany Bovenzi, as well as the CDT students. I would also thank the Indonesian Endowment Fund for Education (LPDP) for the funding support during my studies in Edinburgh.

I would like to thank the member of AGORA group, I really enjoyed working with all of you, thanks for all the help in the paper writing, practice talks, or other research discussions. I would also like to thank the EdinburghNLP group. I am very fortunate to be part of such a big research group which always give constructive feedback to each other. I thank Avashna, Irene, Lucia, Mona, Spandana for being great office-mates. Thanks to Annie, Duygu, Federico, Gozde, Joana, Jonathan, Maria, Naomi, Nick, Pippa, Sorcha, Toms, Yevgen, and other ILCC members for being great friends during my stay in Edinburgh. Many thanks to my Indonesian friends in Edinburgh, Danny, Febrian, Krisna, Lala, Rida, Rozi, and Viona. Special thanks to Ida, Marco, and Sameer for sharing an awesome and crazy friendship that always makes me smile. I will greatly miss our lunch and coffee break times.

Finally, I would like to thank Samuel for always being there for me through my ups and downs. Thank you so much for making me smile, for making me laugh, and always pushing me to go after my dreams. Thank to all my friends and my family, especially my mom for the never-ending support throughout my entire PhD journey. I would not be able to be at this stage without your encouragement and emotional support, and I dedicate this thesis to all of you.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Clara Vania*)

For my mom and Samuel.

Table of Contents

Chapter 1

Introduction

Technology has become an integral part of our daily lives. Every day, people communicate with technology to fulfill their information needs. This includes typing some keywords into a search engine, making a reservation, or playing online music. As technology becomes more advanced, people also aim for a more seamless and natural communication using human languages. The ultimate goal of natural language processing (NLP) is to understand human languages—either written or spoken—to allow communication between humans and computer.

But human languages are remarkably complex and diverse. Consider the following English sentence:

Mary baked a chocolate cake for her mom's birthday.

Though relatively short, this sentence has a complex meaning. It has the basic *who-did-what-to-whom* information, that is, 'Mary baked a cake', as well as more detailed information: the cake is a chocolate cake and Mary baked it to celebrate her mom's birthday.

To understand the meaning of a sentence, we must first understand the meaning of its individual words. However, words themselves are also composed of even smaller meaningful units called **morphemes**. For instance, the word 'baked' is actually composed of two morphemes, 'bake' and '-ed', where the latter represents an action happened in the past. It is different from present action, which is indicated by morpheme'-s', as seen in 'bakes'. These examples of 'baked' and 'bakes' show the *compositional* structure of words: the meaning of a word is a function of the meaning of its morphemes.

One important distinction in morphology is that of **form** and **function** (Tsarfaty and Sima'an, 2008). Form concerns with the phonetic realization—the actual form in speech or writing—of a morpheme, while function concerns with the meaning that a morpheme bears. For instance, English plural morpheme can be realized with an '-s' suffix as in 'cats' or an '-es' suffix as in 'churches'. Sometimes a form can also be ambiguous; the '-s' suffix can both express plural meaning on nouns or third person singular on verbs ('gives' or 'takes'). This many-to-many relation between form and function makes modeling morphology challenging for NLP.

Modeling morphemes is important for many NLP tasks because languages are *productive*. People are continuously inventing new words and applying morphemes to them. When 'selfie' was invented, people naturally use 'selfies' as its plural form. Similarly, people come up with 'binge-watch' and can easily understand its morphologically variants 'binge-watching' or 'binge-watched'. Like humans, we expect NLP systems to understand the morphological (word formation) processes in the language. When a user types 'cat' in a search engine, he or she would expect to get all documents containing both 'cat' and 'cats'. When translating 'cats' to another language, we expect our translation system to understand that it is a plural from and generate the correct translation in the target language. If the target language is Indonesian, this means generating 'kucing-kucing' (*more than one cat*) instead of 'kucing' (*one cat*). In this example, Indonesian applies *reduplication* for plural forms; other languages might use other morphological processes to form words.

Morphology in NLP. Until quite recently, statistical models of NLP generally modeled individual word forms as atomic units. Ignoring the fact that words are composed of morphemes introduces limitations to these models. First, they make a closed vocabulary assumption, enabling only generic out-of-vocabulary (OOV) handling. Second, they cannot exploit systematic functional relationships in learning. For example, 'cat' and 'cats' stand in the same morphological relationship as 'dog' and 'dogs'. While this relationship might be discovered for these specific frequent words, there is no guarantee that they can learn the same effect on the much rarer words 'sloth' and 'sloths'.

Traditional models address these problems by incorporating relevant morphological analyses (present/past tense, singular/plural, etc) as features (Koehn and Hoang, 2007; Avramidis and Koehn, 2008; Tsarfaty et al., 2010). However, morphological analyses require expensive morphological annotations and even automatic analyzers require annotated data for training.

While a single morpheme can have multiple realizations, they are usually similar in forms. Thus, the general relationship between form and function can be exploited by identifying similar subword units (characters, character n-grams, or word segments) across words. Indeed, models based on subword units are now used in many NLP systems as neural networks became prevalent a few years ago. These so-called subword unit models represent words by composing representations of subwords. Thus, instead of learning representations for each possible words, they learn representations of each possible subwords. Since the number of possible subwords is a lot smaller than the number of possible words, subword unit models usually have fewer number of parameters compared to the word-level model. They also can represent rare and OOV words, and they do not require expensive morphological analyzers.

This thesis investigates whether subword unit models, particulary character-level models, learn morphology. Empirically, character-level models (Ling et al., 2015a; Kim et al., 2016) have been quite successful, and this has given rise to some *untested* claims:

- Ling et al. (2015a) claim that character-level models can "model the complex form-function relationship, which captures non-compositional effects, where small orthographic differences may correspond to large semantic or syntactic differences (*butter* vs. *batter*) ... in addition to the more regular effects due to, e.g., morphological processes."
- Lee et al. (2017) claim that character-level models "can easily be applied to a multilingual translation setting. Between European languages where the majority of language alphabets overlaps, a character-level model may easily identify morphemes that are shared across different languages."

These claims strongly imply that character-level models learn morphology, and they have raised a question and debate about explicit modeling of morphology in NLP. Ling et al. (2015b) propose that "prior information regarding morphology ... among others, should be incorporated" into character-level models, while Chung et al. (2016) counter that it is "unnecessary to consider these prior information" when modeling characters. Whether we need to explicitly model morphology is a question whose answer has a real cost: as Ballesteros et al. (2015) note, morphological annotation is expensive, and this expense could be reinvested elsewhere if the predictive aspects of morphology are learnable from strings.

Morphology varies across languages. Character-level models are fairly new and early studies (Ling et al., 2015a; Kim et al., 2016) tested them on just a few languages, mainly English and some Indo-European languages. As alluded to earlier, languages vary with respect their morphological processes. In the morphological typology spectrum, English is closer to the *analytic* side since it uses relatively fewer morphemes per word. Some grammatical functions (tenses or number) are expressed using morphemes, but there are many other functions which are conveyed by function words ('the' or 'in') or word order (subject-verb-object). Other *synthetic* languages which have many morphemes per word are situated towards the other side of the spectrum. Since these languages express grammatical functions mostly through morphemes, they tend to have flexible word order. For example, in Czech, *škola* and *školu* are used to express the word 'school' as a subject and an object, respectively. Another example of synthetic languages is Turkish, which is well-known to have many (five or more) morphemes 'glued' together in a single word, resulting in a quite long word form (Kornfilt, 1997):

(1)	*oku*	*+r*	*+sa*	*+m*
	read	+AOR	+COND	+1SG
	'If I read ...'			

(2)	*okú*	*+ya*	*+ma*	*+yabil*	*+ir*	*+im*
	read	+POT	+NEG	+POT	+AOR	+1SG
	'I might not be able to read'					

As stated by Bender (2013), languages are typologically diverse, and the behavior of a model on one language may not generalize to others. How do character-level models interact with languages of different morphological typologies? **This thesis shows that character-level models are effective across typologically diverse languages, but prior knowledge of morphology is still important for neural models.**

1.1 Thesis Outline

This thesis investigates how subword unit representation models learn morphology, how they interact with typologically diverse languages, and how they perform under low-resource NLP settings. This thesis is organized as follows:

Chapter 2 This chapter reviews background materials of this thesis. We present some key concepts of morphology and how they relate to syntax. We also discuss the *morphological typology*, one of the crucial components of this thesis to give perspective on how languages vary with respect to their morphological processes. After that, we give a brief overview on how words (and morphology) are represented in NLP models. In the last part of the chapter, we review two recent subword unit representation models which are now used in many NLP models.

Chapter 3 To provide a solid baseline, we systematically compare subword units based representation models, investigating (1) the choice of subword unit, (2) model architecture, and (3) their interaction with language typology. We experiment on language modeling—the task of predicting the next word, given the identities of previous words—which plays a central role in many NLP systems. We show that character-level models are the most effective across ten typologically diverse languages. Character-level models learn orthographic similarity of words—including the ones which are consistent to morphology—so they can better represent rare and OOV words. To investigate our main question, we also provide oracle experiments, in which we compare the predictive accuracy of the character-level models to oracle with access to human annotated morphological information. We show that character-level models are effective across typologies, but their performance still underperform the oracle, even with an order of magnitude more data. This work has been published in Vania and Lopez (2017).

Chapter 4 Our oracle experiments in Chapter 3 suggest that character-level models do not entirely capture morphology. Which aspects of morphology are not captured by these models? We further investigate this question by directly comparing the performance of character-level models with the oracle on dependency parsing task. Dependency parsing aims to extract relationship between words in a sentence (such as subject or object), which in some languages, interacts with morphology. We employ parsing models with character-level input and experiment on twelve typologically diverse languages. First, we test explicitly why character-level models are better than standard word-level models. We then demonstrate how our parsing models can benefit from explicit morphological modeling for *case syncretism*, a linguistic phenomena where different morphological cases are expressed using the same word form[1]. This study suggests that our neural models probably do not need all prior linguistic knowledge, but they clearly benefit from some knowledge in addition to raw text input. This work is published in Vania et al. (2018) and Vania and Lopez (2018).

Chapter 5 Annotated data are available for only a tiny fraction of the world's languages. So far, our character-level models are trained using a relatively fair amount of data and it is not understood how they perform on resource-poor conditions. In this chapter, we examine a set of simple strategies to improve low-resource dependency parsing which exploit small linguistic annotations and raw text information: (1) data augmentation, (2) cross-lingual training, and (3) transliteration. This also allows us to test the claim whether character-level models can identify morphemes that are shared across different languages (Lee et al., 2017). We apply parsing models which take character-level input and experiment on three truly low-resource languages: North Sami, Galician, and Kazakh. We show how our strategies, particularly cross-lingual training helps low-resource parsing. However, our analysis suggests that cross-lingual training helps mostly because of the structural similarities between languages and we do not yet find any strong evidence of morphological transfer in the cross-lingual training.

Chapter 6 The summary of our main findings and pointers for future research directions.

[1]We discuss morphological case more detail in Chapter 2.

Chapter 2

Background

This chapter introduces background material which sets the fundamental building block of this thesis. We begin by reviewing some key aspects of morphology, its relation to syntax, and to morphological typology (Section 2.1). We then discuss the language modeling (Section 2.2) and how it can be used for learning word representations (Section 2.3). Finally, we discuss word representation models based on subword units, and how they might help downstream task which benefits from morphological information such as dependency parsing (Section 2.4).

2.1 Morphology

Morphology is the subfield of linguistics which studies the internal structure of words and how they are formed. In Chapter 1, we have used the notion of 'word' in a quite general way, but it is important to note that the 'word' which can be realized into various word forms is actually called a **lexeme**. A lexeme is an abstract sense of word and we can think of it as a lexical entry in a dictionary. A **lemma** is the canonical form that is used to represent the lexeme. For example, the English dictionary locates 'write', 'writes', 'written', and 'writing' all within the same lexeme, with WRITE as the lemma. A *concrete* word observed in a piece of text is known as a **word token**. A **word type** is each *distinct* word form in a piece of text. For example, 'the cats and the butterflies' has five word tokens and four word types ('the', 'cats', 'and', 'butterflies'). A morphological **paradigm** defines a complete set of word forms that belongs to the same lexeme. Table 2.1 shows the paradigm of the Latin noun lexeme INSULA.

	SINGULAR	PLURAL
NOMINATIVE	*insula*	*insulae*
ACCUSATIVE	*insulam*	*insulās*
GENITIVE	*insulae*	*insulārum*
DATIVE	*insulae*	*insulīs*
ABBLATIVE	*insulā*	*insulīs*

Table 2.1: The paradigm of the Latin noun lexeme INSULA (island). (Haspelmath, 2010)

A fundamental part of morphology is the correspondence between **form** (structure) and **function** (meaning). A **morpheme** is the smallest meaningful unit in the grammar of a language; it carries a specific function that contributes to meaning. A **morph** is the phonetic realization—the actual form in speech or writing—of a morpheme, which may vary from word to word. For instance, the English plural morpheme *-s* can be realized with an '-s' as in 'dogs' or an '-es' as in 'branches'.[1] These different realizations of a single morpheme is called **allomorphs**. A **null morpheme** is a morpheme that has no phonetic form. An example is the present tense of English for non third person singular ('*I* sing' vs. '*They* sing'). The mapping from form to function is not always straightforward and often requires contextual information for interpretation (Beard, 1988; Anderson, 1992; Tsarfaty and Sima'an, 2008).

Word forms belong to the same paradigm typically share at least one core morpheme with a concrete meaning and additional shorter morphemes called **affixes**. A **base** is the core morpheme(s) that an affix is attached to. A **root** is a base which cannot be broken up any further into component morphemes. For example, in 'sustainability', *sustainable* is the base and *sustain* is the root. As such, the notion of base is determined by the notion of affix (Haspelmath, 2010). An affix expresses one or more dependent features of the core meaning, such as person, gender, or tense. For example, 'tries' consists of a base *try* and a dependent morpheme *-s* representing a set of third person, singular, present tense features. A **morphological analysis** identifies the lemma, the part of speech (POS), and the set of features of a word. These distinctions are summarized in Table 2.2.

Affixes can further be divided into several types according to their position within the

[1] Note that morpheme does not have actual realization. For clarity, this thesis uses italic to denote morpheme (*-s*) and quotes to denote morph ('-es').

word	*tries*
morphemes	*try* + *-s*
lemma	*try*
affix	*-s*
morphs	‘tri’ + ‘-es’
morphological analysis	*try*+VB+3RD.SG.PRES

Table 2.2: The morphemes, morphs, and morphological analysis of *tries*.

Type	Example		
	Affix	Lemma	Word form
prefix	*peN-*	*ajar* (teach)	*pengajar* (teacher)
infix	*-el-*	*tunjuk* (point)	*telunjuk* (index finger)
suffix	*-an*	*makan* (eat)	*makanan* (food)
circumfix	*peN-...-an*	*buka* (open)	*pembukaan* (opening)

Table 2.3: Example of different affix types in Indonesian.

word. Affixes that follow the base are called **suffixes** and affixes that precede it are called **prefixes**. When an affix consists of both suffix and prefix, it is called **circumfix**. Some languages also have **infixes**, which are affixes that are inserted inside the base. The attachment of affixes might change the base form, depending on the phonemes of the base and the affixes. For example, the Indonesian prefix *peN-* becomes *peng-* when the base form starts with a vowel and becomes *pem-* when the base form starts with a /b/. A full example of different affix types is given in Table 2.3.

2.1.1 Inflection and Derivation

Tables 2.2 and 2.3 show how morphemes can add different kinds of meanings to the core meaning represented in the base form. According to the principal of word building processes, morphemes can be divided into two categories, namely **inflectional** and **derivational** morphemes. Inflection modifies the base so that it can fit into a particular syntactic slot in a sentence; it does not change the word-class of the base. In contrast, derivation may alter the meaning of the the base and create a new lexeme from a base (Katamba and Stonham, 2006). Table 2.4 gives examples of inflectional and derivational processes in English.

Process	Base	Word-class	New form	Word-class	Function/Meaning
Inflection	*cat*	NOUN	*cats*	NOUN	marks plural number
	read	VERB	*reads*	VERB	marks present tense, singular person
Derivation	*kind*	ADJ	*unkind*	ADJ	'not' kind
	write	VERB	*writer*	NOUN	an agent who writes

Table 2.4: Inflections and derivations in English.

This thesis mostly deals with inflectional morphology. To provide some background on inflectional morphology this section discusses some properties of inflection. First, we provide definitions of some inflectional features which are quite common in languages with rich morphology. After that, we discuss the relevance of inflectional morphology to the syntax.

2.1.1.1 Inflectional features

The degree of inflection varies across languages. An English verb can have around six inflected forms, while a Polish verb may have up to 100 inflected forms (Janecki, 2000). The number of forms that a lexeme can have depends on the number of possible inflectional **features** and **values** in the language. Table 2.5 shows morphological features and values that are common across languages.

Number feature expresses quantity. It typically takes two values (singular or plural) but a few languages also use *dual* value for number. For example, in Slovenian, *volk* ('wolf') has forms *volk* ('one wolf'), *volka* ('two wolves'), and *volkovi* ('more than two wolves'). This feature is usually marked on nouns, verbs, and sometimes adjectives.

Case is a system to indicate the semantic and syntactic role of a word in a sentence. Typically, case marks the relationship of a noun to a verb, preposition, postposition, or another noun at a clause or phrase level (Iggesen, 2013). Examples of case values are: *nominative* for subjects, *accusative* for direct objects, *dative* for indirect objects, and *genitive* for possession or relationships with another noun. In languages with extensive case systems, case is also used to mark other roles such as location (*locative*) or tool

Feature	Value	On word category
Number	SINGULAR, PLURAL, ...	NOUN, PRON, VERB, ADJ, ADP
Case	NOMINATIVE, ACCUSATIVE, ...	NOUN, PRON, ADJ, ADP
Gender	MASCULINE, FEMININE, ...	NOUN, PRON, ADJ, ADP
Person	1ST, 2ND, 3RD, ...	NOUN, PRON, VERB, ADJ, ADP
Tense	PAST, PRESENT, FUTURE, ...	VERB
Aspect	IMPERFECT, PERFECT, ...	VERB
Mood	INDICATIVE, IMPERATIVE, ...	VERB

Table 2.5: Common inflectional features and values across languages (Haspelmath, 2010).

(*instrument*). This feature is mostly prominent in the Indo-European languages, in particular the Slavic languages. The following are examples of case marking in Russian (3a) and Turkish (3b) for the sentence ‘Mark broke the window with a hammer.’ (Sahin et al., 2019):

(3)	a.	*Mark-∅*	*razbi-l-∅*	*okn-o*	*molotk-om*
		Mark-NOM.SG	break-PST-SG.M	window-ACC.SG	hammer-INST.SG
		‘Mark broke the window with a hammer.’			
	b.	*Mark-∅*	*pencere-yi*	*çekiç-le*	*kr-d*
		Mark-NOM.SG	window-ACC.SG	hammer-INST.SG	break-PAST.3.SG
		‘Mark broke the window with a hammer.’			

In those examples, both Russian and Turkish use accusative case (ACC) and instrument case (INST) markers for the object ‘window’ and the instrument tools ‘hammer’, respectively. Case is mostly marked on nouns, pronouns, and adjectives.

Gender is a specific feature of nouns which usually form agreement with other categories such as adjectives, pronouns, articles, or verbs. For example, in Spanish, the form of the article depends on the noun that it modifies. Spanish distinguishes *masculine* and *feminine* genders. For example, in singular nouns, it uses articles *el* (masculine) and *la* (feminine): *el gato* and *la gata* (‘the cat’). A few other languages like Icelandic and German use three genders, and some languages like Bantu might have up to 7-10 genders (Corbett, 2013). The combination of gender, case, and number is

	MASCULINE		FEMININE	
	SINGULAR	PLURAL	SINGULAR	PLURAL
NOMINATIVE	*der Tisch*	*die Tische*	*die Kraft*	*die Kräfte*
ACCUSATIVE	*den Tisch*	*die Tische*	*die Kraft*	*die Kräfte*
DATIVE	*dem Tisch(e)*	*den Tishcen*	*der Kraft*	*den Kräften*
GENITIVE	*des Tisch(e)s*	*der Tische*	*der Kraft*	*der Kräfte*

Table 2.6: Examples of German noun inflection (declension) for *der Tisch* (the table) and *die Kraft* (the power).

typically used to generate the correct inflection, as seen in Table 2.6.[2]

Next, we discuss features which are common on verbs. **Person** indicates the subject of a verb, whether it is the speaker (*1st*), the addressee (*2nd*), or a third party (*3rd*). **Tense** marks the temporal information of verbs, whether it is in the *past*, *present*, or *future*. In English, person and tense are usually combined with number to inflect verbs. For example, the *-s* morpheme on verb is used to represent 3rd person, singular, present tense features.

Aspect refers to the internal temporal constituency of an event, whether the action is completed or will be completed (*perfective*); non-completed (*imperfective*); or repetitive (*habitual*). For example, Polish verb *pije* ('to drink') for a third person singular can be realized into the imperfective form *pije* ('is drinking') or the perfective form *wypije* ('will drink'). **Mood** marks the speaker's point of view. Typical values for mood are *indicative* (objective facts), *imperative* (commands), and *subjunctive* (non-realized events). Some languages, including English and many Indo-European languages conflate tense, mood, and aspect in a single morpheme. For example, the morpheme '-s' in 'sings' expresses both present tense and indicative mood. English uses actual verb form for imperative mood: 'Sing!' or 'Go!'. On the other hand, Italian marks imperative mood through morphemes: for verbs that end with *-are*, like *cantare* ('to sing') some of the possible forms are *lei canti!* (singular) or *loro cantino!* (plural).

[2]https://en.wikipedia.org/wiki/German_grammar#Case (last access: January 2019)

2.1.1.2 Syntactic agreement and government

How does inflectional morphology relate to syntax? In Chapters 4 and 5, we will examine the performance of subword units representation models on dependency parsing—a syntactic task which benefits substantially from morphological knowledge. To provide some background, here we discuss two syntactic relations which interact with morphology, namely syntactic agreement and syntactic government.

Syntactic agreement This relation deals with how inflectional value of a given word or phrase must be the same or *agree* with another word or phrase in a sentence. The first is called as the *target* while the latter is called as the *controller*. One example is the English subject-verb agreement, where a verb (the target) must agree with its subject (the controller) in number value. In languages with richer morphology, syntactic agreement may involve multiple features and/or multiple controllers. An example is in Yimas, an endangered language in Papua New Guinea, where a verb should agree with subject and object in person, gender, and number values (Foley, 1991):

(4)

Krayŋ	*narmaŋ*	*k-n-tay.*
frog.SG(G6)	woman.SG(G2)	3SG.G6.P-3SG.G2.AG-SEE

‘The woman saw the frog.’

In this example, the verb *tay* (‘see’) is inflected so that it agrees with both the subject and the object in number. If we relate back to Table 2.5, the common inflectional features of nouns and pronouns are the same as adjectives and adpositions. This is because the controller of agreement relations are usually nouns, pronouns, or noun phrases while the targets are usually adjectives, adpositions, or determiners. Verbs are also frequent targets for noun agreement, especially in person, number, and sometimes gender (Haspelmath, 2010).

Syntactic government Unlike syntactic agreement in which the controller and the target should have the *same* inflectional value, in syntactic government, the target might have a different inflectional feature/value with the controller. Syntactic government introduces a non-symmetrical (dependency) relation between two words w_1 and w_2, which can be written as $w_1 \rightarrow w_2$. Here, w_1 is said to be the *governor* or *head* of w_2, while w_2 is the *dependent* or *child* of w_1. In languages with case marking, syntactic government typically involves governing a word to have a particular grammatical case feature, also called as **case government**. In this relation, usually a verb or a preposition will govern the grammatical case on its noun dependents. For example, the presence of negation on a Polish verb may lead to the different cases on it's direct object; a negated Polish verb requires a genitive case while a typical, non-negated Polish verb requires an accussative case, as seen in the examples below (Patejuk and Przepiórkowski, 2014):

(5) a. *Poczytam ksik.*
read.1st.SG book.ACC
'I'll read a book'

b. *Nie poczytaj ksiki czy gazety.*
NEG read.3rd.PL book.GEN or newspaper.GEN
'They won't read a book or a newspaper.'

2.1.2 Morphological Typology

Languages vary considerably in their morphological complexity. Morphological typology classifies languages based on the processes by which morphemes are composed to form words. Linguists consider the following categories of languages according to their *dominant* morphological processes (Katamba and Stonham, 2006):

Analytic or isolating languages are languages with very little degree of inflection. To convey meaning, these languages depend on a strict word order or function words, such as prepositions, postpositions, articles, etc. Analytic languages are mostly found in East Asia, South East Asia, West Africa, and South Africa. The opposite of analytic languages are **synthetic languages**, where inflections are used to express grammatical properties. Examples 6 and 7 show the distinction between Vietnamese, a highly analytic (**isolating**) language with West Greenlandic, a highly synthetic (or **polysynthetic**) language.

(6) *Hai ú.a bo? nhau là ti gia-inh thàng chông.*
two individual leave each.other be because.of family guy husband.
'They divorced because of his family'

(Nguyen, 1997)

(7) *Paasi-nngil-luinnar-para illa-juma-sutit.*
understand-not-completely-1SG.SBJ.3SG.OBJ.IND come-want-2SG.PTCP
'I didn't understand at all that you wanted to come along.'

(Fortescue, 1984)

It is important to note that the distinction between analytic and synthetic is not bipartition, but rather continuum, ranging from the most analytic (no inflection) to the most (poly)synthetic languages (Haspelmath, 2010). For example, although English has more morphology than Vietnamese, it has a lot less morphology than many other languages, like Czech, Russian, Turkish, or Arabic. While there is no fixed measurement for degree of synthesis, typically we assume languages which use a lot more morphology than English as highly synthetic. Some literature also refer to these languages as **morphologically rich** or **morphologically complex** languages.

Fusional languages are languages which realize multiple inflectional features in a single morpheme. For example, English verbs express number, person, and tense in a single affix. For example, the morpheme '-s' in 'wants' represents third person singular present tense (3RD.SG.PRES) features. Fusional process are commonly found in most of Indo-European languages: Punjabi, Greek, Italian, German, Icelandic, all Baltic and Slavic languages, among others.

Agglutinative languages are languages with a more one-to-one mapping between inflectional features and morphemes. Morphemes are concatenated to form a word and the boundaries between morphemes are mostly clear. In theory, the concatenation process can add an infinite number of morphemes and therefore languages with agglutinative process typically have quite long words. Example of agglutinative languages are Turkish, Finnish, and Hungarian. Below is an example of agglutinative process in Turkish:

(8) *oku-mal-y-m-z*
read-NEC-BE-REP.PST-1PL
'They say that we have to read'

(Kornfilt, 1997)

Templatic languages (also called **root and pattern** morphology) form words by inserting consonants and vowels into a consonantal root based on a given pattern. This *non-concatenative* process is mainly found in Semitic languages, such as Arabic and

Hebrew. While this particular morphological process has its own challenges since it is quite different with other well-studied languages, there is also additional difficulty as the available written text often do not exhibit the process. For example, Arabic is typically written **unvocalized** (without the vowels) which often leads to one form have different meanings. Table 2.7 gives examples of Arabic verbs derived from the root morpheme 'ktb' ('notion of writing')

Pattern	**Verb stem**	**Gloss**
$C_1\mathrm{a}C_2\mathrm{a}C_3$	*katab*	'wrote'
$C_1\mathrm{a}C_2C_2\mathrm{a}C_3$	*kattab*	'caused to write'
$C_1\mathrm{aa}C_2\mathrm{a}C_3$	*kaatab*	'corresponded'
$\mathrm{a}C_1C_2\mathrm{a}C_3$	*aktab*	'caused to write'
$\mathrm{ta}C_1\mathrm{aa}C_2\mathrm{a}C_3$	*takaatab*	'wrote to each other'

Table 2.7: Examples of Arabic verbs derived from root morpheme *ktb*. Each C_i refers to the consonant in position i (Roark and Sproat, 2007).

Reduplication is another kind of non-concatenative process where a word form is produced by repeating part or all of the root to express new features. This process is very common in Australia, South Asia, Austronesian, and many parts of Africa (Rubino, 2013). Table 2.8 shows an example of reduplication in Indonesian.

Root		**Word form**	
anak	'child'	*anak-anak*	'children'
buah	'fruit'	*buah-buahan*	'various fruits'
mobil	'car'	*mobil-mobilan*	'a toy car'
tolong	'help'	*tolong-menolong*	'help each other'
merah	'red'	*kemerah-merahan*	'reddish'

Table 2.8: Full and partial reduplication in Indonesian.

We have just shown how morphology is expressed differently across languages, and this has an implication that NLP models must account for all of these different typologies to process languages. In the next subsections, we will discuss how words and morphology are actually modeled in NLP. For consistency, we will use mathematical notation listed in Table 2.9 to described models that we used throughout this thesis.

Notation	Description
x	a unit (word, morpheme, character, etc)
$x = u_1, \ldots, u_{\lvert x \rvert}$	decomposition of x into a sequence of $\lvert x \rvert$ units
$\lvert x \rvert$	size/length of x
$\mathbf{1}_x$	one-hot vector for x
$\mathbf{x}$	a column vector
$\mathbf{x}^{\mathrm{T}}$	transpose of $\mathbf{x}$
$\mathbf{x}_1, \ldots, \mathbf{x}_n$	a sequence of vectors
$[\mathbf{x}; \mathbf{y}]$	concatenation of $\mathbf{x}$ and $\mathbf{y}$
$\mathbf{X}$	a matrix
$\mathbf{X}^{\mathrm{T}}$	transpose of $\mathbf{X}$

Table 2.9: Notation used throughout this thesis.

2.2 Language Modeling

We start our discussion by reviewing language modeling, which is the task of assigning the probability of a sentence. Language model plays a central role in many NLP tasks such as machine translation or speech recognition to assess the grammaticality and fluency of a text or utterance.

Let $w = w_0, \ldots, w_T$ be a sequence of words in a sentence. We assume that each sentence starts with a Begin-Of-Sentence (BOS) token and ends with an End-Of-Sentence (EOS) token. Thus, in our notation $w_0 = \text{BOS}$ and $w_T = \text{EOS}$. Adding BOS and EOS tokens allows us to properly model the probability of a word being the first or the last word in a sentence. A language model computes the probability of w as:

$$P(w) = \prod_{t=0}^{T} P(w_t \mid w_0, \ldots, w_{t-1}) \tag{2.1}$$

That is, the probability of a sentence is a product of probabilities of individual words, each conditioned on the **history** of previous words in the sequence.

Recurrent Neural Network Language Model Mikolov et al. (2010) propose a language model based on a recurrent neural network (RNN; Hochreiter, 1991), which, in theory, can take infinite amount of context and thus can directly model Equation 2.1.

As can be referred from the term *recurrent*, an RNN language model assumes a sequence of inputs and processes them one by one in a successive order. Because inputs are processed in this way, it is common to refer to the order of the sequence as *time steps* $t \in [0,N]$, where $N+1$ is the length of the entire history. At each time step t, an RNN computes the following:

$$\mathbf{h}_t = g(\mathbf{U}^T \cdot \mathbf{w}_t + \mathbf{H}^T \cdot \mathbf{h}_{t-1}) \tag{2.2}$$

$$\hat{w}_{t+1} \sim \text{softmax}(\mathbf{V}^T \cdot \mathbf{h}_t) \tag{2.3}$$

where $\mathbf{w}_t \in \mathbb{R}^d$ is a continuous representation (or, equivalently an **embedding**) of the current word, $\hat{w}_{t+1}$ is the predicted target word that is drawn from the distribution over all the possible target words, $\mathbf{U} \in \mathbb{R}^{h \times d}$, $\mathbf{H} \in \mathbb{R}^{h \times h}$, $\mathbf{V} \in \mathbb{R}^{|V| \times h}$ are model parameters that are learned during training, d and h are the dimension of the word representation and hidden layer, respesctively. Note that, since $\mathbf{h}_t$ is computed recursively from $\mathbf{h}_1, \ldots, \mathbf{h}_{t-1}$, an RNN language model can represent the entire history of previous words.

Although in theory RNN should be able to store the entire history, in practice when it processes a very long sequence, earlier information tends to be forgotten (Hochreiter, 1991). To address this, variants of RNN such as Long Short Term Memory (LSTM; Hochreiter and Schmidhuber, 1997) or Gated Recurrent Unit (GRU; Cho et al., 2014) are often preferred over the vanilla RNN. While the specific computations are different, these models take the same inputs, outputs, and recursive form of Equations 2.2 and 2.3.

The standard evaluation metric to measure the quality of a language models is called **perplexity**. Given a held-out *test data* of length N, we calculate the perplexity of a language model using the average cross-entropy H:

$$\text{perplexity} = 2^H \tag{2.4}$$

where H is defined as:

$$H = \frac{1}{N} \sum_{i=1}^{N} -\log_2 P(w_i | w_0, \ldots, w_{i-1}) \tag{2.5}$$

In other words, Equation 2.5 computes the average negative log probability that our language model assigns to each word in our test data. A good quality language model assigns higher probability to an actual sentence that it has never seen before. Thus, the lower perplexity, the better our language model in predicting the next word.

2.3 Learning Word Representations

One main advantage of training a language model using a neural network is that now we can automatically learn representation of words in our data. Recall from Equation 2.2, we use $\mathbf{w}_t$ as the representation of the word token at time t (w_t). How $\mathbf{w}_t$ is computed? It could be simply a learned vector representation of w_t, in which case it is a lookup table or **word embeddings**. These representations are actually just parameters of our language model; they are learned and optimized such that they can best 'fit' our data. In fact, most of NLP models which are based on neural networks can also learn such representations. The key idea is that the learned representation is influenced both by the previous history ($\mathbf{h}_t$), and by the next words (w_{t+1}), so this means the representation is influenced by context. As a result, words with similar context will have similar representations. This idea is similar to the distributional hypothesis (Harris, 1954; Firth, 1957) that inspired studies on traditional count-based word representations (Turney and Pantel, 2010; Landauer and Dumais, 1997).

Although technically we can learn word representations directly on the NLP task itself, language model and its variants (Collobert and Weston, 2008; Mikolov et al., 2010, 2013; Peters et al., 2018) are still the most popular task to learn word representations. This is because they can be trained on a very large, unannotated corpus, allowing us to include as much contextual information as possible during learning. The learned word representations can then be used to initialized word representations (hence called **pre-trained word embeddings**) on the main NLP task that we are interested in. For example, imagine a POS tagging model similar to our previous RNN language model, where instead of predicting the next word $\hat{w}_{t+1}$, we are now predicting the POS of the input word at each time step. We can initialized the word lookup $\mathbf{w}_t$ with a representation learned using language model rather than initialize it randomly.

Continuous word representations have almost become *de facto* input layer in many NLP tasks. The reason on why they perform so well on many tasks is still not very well understood, but one possible explanation is because the learned representations can capture the syntactic regularities ('big' vs. 'bigger') as well as the semantic regularities ('big' vs. 'small') among words (Mikolov et al., 2013). Efforts on understanding word representations were also reflected on various recent workshops such as Representation Learning for NLP (Rep4NLP[3]) and Analyzing and Interpreting Neural Networks for

[3]https://sites.google.com/view/repl4nlp2019

NLP (BlackBoxNLP[4]) workshops. In this thesis, we specifically aim to understand the morphological aspect of word representations.

2.3.1 Subword-Unit Based Word Representation Models

Despite their usefulness, standard word representation models still model words as their atomic units. In Chapter 1, we discussed how this approach introduces limitations to NLP models: it makes a closed vocabulary assumption, enabling only generic OOV handling (using a generic "unknown" vector) and it can not exploit systematic functional relationships in learning. To address these, recent work proposes to replace lookup of $\mathbf{w}_t$ with a function that composes representations of smaller units, such as morphemes, morphs, character n-grams, or characters. In most of these cases, the vocabulary is finite, which makes it possible to represent any word in the language. Moreover, the use of subword-level information means that now words with similar forms, either related morphologically ('cat' vs. 'cats') or not ('best' vs. 'bet') can share similar representations. This also means that rare and unseen words will have more informative representations, especially when their morphologically variants are seen during training. Next, we discuss each of these representations and how they are used in NLP models.

Morphemes and Morphs When human-annotated datasets are available, morphological information can be incorporated into NLP models as features. Traditional count-based vector models (Alexandrescu and Kirchhoff, 2006; Lazaridou et al., 2013) typically treat each morphological feature/morph as a feature and compute word vector representation using vector operations such as concatenation, addition, or multiplication on the feature vectors. For neural models, Cotterell and Schütze (2015) augment a neural language model with a multi-task objective, where a model is trained to jointly predict the next word along with the next morphological feature.

In scenarios where there exists no human annotation, some studies have used morphological analyzers to obtain tags or morphs. The morphological analyzer can be learned via unsupervised methods (Creutz and Lagus, 2002) given text data as input, or trained using supervised learning methods on a small amount of labeled data, which is then used to predict tags for the remaining unlabeled data (Kohonen et al., 2010).

[4]https://blackboxnlp.github.io/

For morphological segmentation where the goal is to approximate morphs (‘stopping’ $\rightarrow$ ‘stopp+ing’), there are two algorithms that have been widely used in neural NLP models: Morfessor (Creutz and Lagus, 2007; Virpioja et al., 2011) and Byte Pair Encoding (BPE; Gage, 1994). Morfessor segments words by recursively splitting words to minimize an objective based on minimum description length (MDL) principle. Early versions were entirely unsupervised, but the most recent version, Morfessor 2.0 (Smit et al., 2014) also includes semi-supervised methods for utilizing annotated data. Luong et al. (2013) embed morphs obtained from Morfessor and compose word representation using recursive neural networks (Socher et al., 2011). Similarly, Botha and Blunsom (2014) and Qiu et al. (2014) embed morphs as well as the original word form and sum their embeddings to produce the final word representation.

BPE is a data compression technique adapted by Sennrich et al. (2016) for word segmentation. It works by iteratively replacing frequent pairs of characters (‘a’, ‘b’) with single unused character (‘ab’), which represent a character n-gram. Eventually, frequent character n-grams (or frequent words) would each be represented by a symbol while rare words would be segmented into their frequent character n-grams. BPE has a single parameter which is the number of merge operations. The final vocabulary size is the size of initial vocabulary (the number of distinct characters) plus the number of merge operations. Sennrich et al. (2016) show that BPE is effective for handling rare words in neural machine translation and after that BPE has became a popular choice for representing words in machine translation (Johnson et al., 2017; Bansal et al., 2019). A similar word segmentation algorithm (Schuster and Nakajima, 2012) is used by Google’s neural machine translation system (Wu et al., 2016) and BERT pretrained language model (Devlin et al., 2019) to segment word into subword units called “wordpieces”.

Character N-grams Both Morfessor and BPE require training data to learn word segmentation models. A rather simpler approach is to approximate morphs using the word’s overlapping character n-grams (Table 2.10). Sperr et al. (2013) represent each word using a binary vector indicating the occurrence of each possible character n-gram in the word ($n \in \{1,2,3,4\}$). Bojanowski et al. (2017) introduce *fastText*, which is a model that represents a word by the sum of its character n-grams vectors ($n \in \{3,4,5,6\}$). Wieting et al. (2016) propose a similar model ($n \in \{2,3,4\}$), but with an additional elementwise non-linearity.

Unit	**Output of $\sigma(\textit{wants})$**
Morfessor	^want, s$
BPE	^w, ants$
char-trigram	^wa, wan, ant, nts ts$
character	^, w, a, n, t, s, $
analysis	want+VB, +3RD, +SG, +PRES

Table 2.10: Morphs produced with Morfessor, BPE, and overlapping character trigrams *wants*.

Characters Morfessor, BPE, and character n-grams (Table 2.10) implicitly assume that morphemes are combined together using a concatenative process and that each morpheme have the same realization on the surface word forms. However, there exist languages with non-concatenative morphological processes (Section 2.1.2), and sometimes there can be changes to the surface morphemes when the morphemes are combined together (‘stop’ + ‘-ing’ $\rightarrow$ ‘stopping’). An alternative option would be to model words from characters instead of morphs. Character-level word representation models do not require explicit word segmentation, instead they rely on neural networks to learn the complex form-function relationship (Ling et al., 2015a; Lee et al., 2017). Santos and Zadrozny (2014) propose a model that jointly use word-level and character-level features to represent words for POS tagging. They apply a convolutional neural network (CNN) to build a vector representation from character-level features and combine it with word-level embedding. Kim et al. (2016) follow a similar approach but use only character-level embeddings. Ling et al. (2015a) employ a different composition model using a bidirectional LSTM over characters to represent inputs for language modeling and POS tagging. These character-level models of Kim et al. (2016) and Ling et al. (2015a) have now been used in many other NLP tasks, especially for processing morphologically complex languages (Lee et al., 2017; Ataman and Federico, 2018; Sahin and Steedman, 2018a). We will discuss each of these models more detail in the next section and examine their relation to morphology in Chapters 3 and 4.

2.3.2 Character-Level Word Representation Models

At the time of the writing, there are many variants of character-level word representation model, but most of them employ either a bidirectional LSTM (biLSTM) or a convolutional neural network (CNN). It is important to note that while these models were originally introduced to model characters, in practice they can be used for any other subword units such as character n-grams or BPE.

Character-Level BiLSTM A natural way to compose a word representation from its characters is to “read" the characters as a sequence. Ling et al. (2015a) introduce a compositional character to word model based on biLSTMs (Graves et al., 2005), which produces a word representation by modeling the complex non-local dependencies between characters in a given word. We call this model **charLSTM**.

Figure 2.1 illustrates the architecture of the model. Let C be the vocabulary, which is the set of all characters in the language. Given a word $w = c_1, c_2, \ldots, c_{|w|}$, each character c_i is represented by a one-hot encoding, which is then mapped into its continuous representation (or **character embedding**), $\mathbf{c}_i$. At each time step i, we feed $\mathbf{c}_i$ into a biLSTM, which consists of an LSTM going over the sequence (**forward LSTM**) and another LSTM going over the reverse input sequence (**backward LSTM**). An LSTM computes the following hidden state for the character at position i:

$$\mathbf{h}_i = \text{LSTM}(\mathbf{c}_i, \mathbf{h}_{i-1}) \tag{2.6}$$

where $\mathbf{h}_{i-1}$ is the hidden state at the previous time step.

Let $\mathbf{h}_n^{fw}$ and $\mathbf{h}_0^{bw}$ be the final hidden states of the forward and backward LSTMs, respectively. We compute the final word representation by adding both final hidden states:

$$\mathbf{w}_t = \mathbf{W}_f \cdot \mathbf{h}_n^{fw} + \mathbf{W}_b \cdot \mathbf{h}_0^{bw} + \mathbf{b} \tag{2.7}$$

where $\mathbf{W}_f$, $\mathbf{W}_b$, and $\mathbf{b}$ are parameter matrices that are learned during training.[5]

Character-Level CNN Kim et al. (2016) propose a model which computes word representations from their characters using a convolutional neural network (CNN). This model, which we refer to as **charCNN**, is illustrated in Figure 2.2.

[5]Another way to combine the forward and backward final LSTM states is by using concatenation.

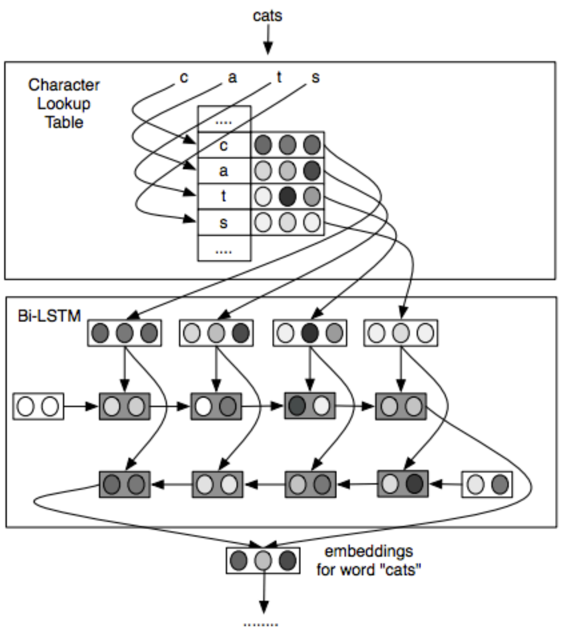


Figure 2.1: Character-level biLSTM model (figure from Ling et al. (2015a)).

Similar to charLSTM, charCNN also uses the set of characters in the language as its vocabulary. Given a word $c = c_1, c_2, \dots, c_k$, we first build a character embedding matrix $\mathbf{C} \in \mathbb{R}^{d \times k}$, where the i-th column corresponds to the embeddings of c_i. We then apply a narrow convolution (without zero padding) between $\mathbf{C}$ and a filter $\mathbf{F} \in \mathbb{R}^{d \times n}$ of width n to obtain a feature map $\mathbf{f} \in \mathbb{R}^{k-n+1}$. In particular, the computation of the j-th element of $\mathbf{f}$ is defined as

$$\mathbf{f}[j] = tanh(\langle \mathbf{C}[:, j : j+n-1], \mathbf{F} \rangle + b) \tag{2.8}$$

where $\langle A, B \rangle$ is a component-wise inner (Hadamard) product and b is a bias. We then calculate the max-over-time of the feature map:

$$y_j = \max_j \mathbf{f}[j] \tag{2.9}$$

We just give an illustration for applying one filter which picks out a character n-gram. In practice, the CNN model applies filters of varying width to representing different length of n-grams. To produce the word representation, we concatenate the max-over-time of each filter, $\mathbf{w}_t = [y_1, \dots, y_m]$, where m is the number of filters applied.

To model the complex interactions between character n-grams picked up by the filters, we can apply a highway network, which allows some dimensions of $\mathbf{w}_t$ to be carried or transformed. An example for one layer of a highway network is as follows:

$$z = \mathbf{t} \odot g(\mathbf{W}_h \cdot \mathbf{w}_t + \mathbf{b}_h) + (\mathbf{1} - \mathbf{t}) \odot \mathbf{w}_t \tag{2.10}$$

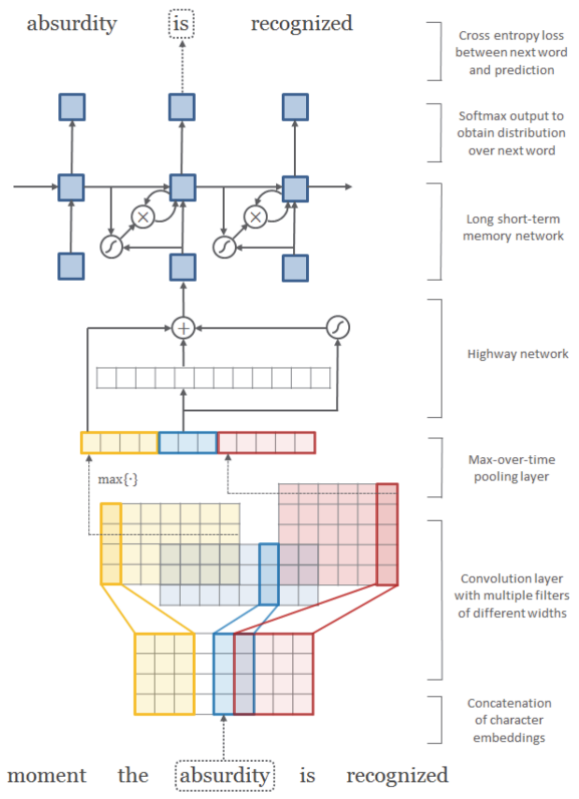

absurdity
is
recognized
Cross entropy loss between next word and prediction
Softmax output to obtain distribution over next word
Long short-term memory network
Highway network
Max-over-time pooling layer
max{·}
Convolution layer with multiple filters of different widths
Concatenation of character embeddings
moment
the
absurdity
is
recognized

Figure 2.2: Character-level CNN model (figure from Kim et al. (2016)).

where g is a non-linearity function, $\mathbf{t} = \sigma(\mathbf{W}_t \cdot \mathbf{w}_t + \mathbf{b}_t)$ is a *transform* gate and $(\mathbf{1} - \mathbf{t})$ is the *carry* gate. $\mathbf{W}_h$ is a weight parameter matrix and $\mathbf{b}_h$ is a bias.

In terms of computational speed, character-level models are significantly slower than the standard word lookup approach. Ling et al. (2015a) report that charLSTM is three times slower than word lookup approach, while for charCNN, Kim et al. (2016) observe that it is twice as slow. With GPUs, charCNN is arguably faster than charLSTM since the convolution operations have been optimized. For charLSTM, the model still needs to perform two LSTMs traversals, which can be slow for longer words. The authors of both models also mention some efficiency techniques to reduce the computation time at the cost of memory, such as caching or pre-computation of the most frequent words.

CharLSTM and charCNN along with BPE were the first models which popularized the idea of open vocabulary model, which can essentially represent any given word in the language, including OOV words. In the original papers, the authors applied their models to language modeling, POS tagging, and machine translation, however in practice these models can be plugged in any neural models as a word representation

Models	Subword Unit(s)	Composition Function	Task
Sperr et al. (2013)	words, character n-grams	addition	language modeling
Luong et al. (2013)	morphs (Morfessor)	recursive NN	word similarity
Botha and Blunsom (2014)	words, morphs (Morfessor)	addition	language modeling
Qiu et al. (2014)	words, morphs (Morfessor)	addition	word similarity, word analogy
Santos and Zadrozny (2014)	words, characters	CNN	POS Tagging
Cotterell and Schütze (2015)	words, morphological analyses	addition	language modeling, morphological similarity
Sennrich et al. (2016)	morphs (BPE)	none	machine translation
Kim et al. (2016)	characters	CNN	language modeling
Ling et al. (2015a)	characters	biLSTM	language modeling, POS tagging
Ballesteros et al. (2015)	characters	biLSTM	dependency parsing
Wieting et al. (2016)	character n-grams	addition	word and sentence similarity, POS tagging
Bojanowski et al. (2017)	character n-grams	addition	word similarity
Vylomova et al. (2017)	characters, morphs (Morfessor)	biLSTM, CNN	machine translation
Miyamoto and Cho (2016)	words, characters	biLSTM	language modeling
Rei et al. (2016)	words, characters	biLSTM	sequence labeling
Lee et al. (2017)	characters	CNN	machine translation
Kann and Schütze (2016)	characters, morphological analyses	none	morphological reinflection
Heigold et al. (2017)	words, characters	biLSTM, CNN	morphological tagging
Bohnet et al. (2018)	words, characters	meta biLSTM	morphosyntactic tagging

Table 2.11: Previous work on representing words through compositions of subword units.

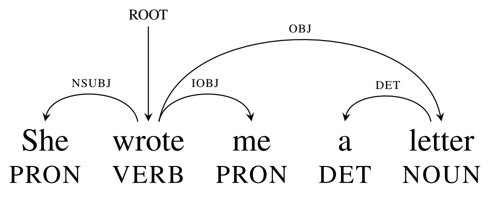


Figure 2.3: A *projective* dependency graph.

module or **word encoders** (§2.3.1). While they have shown to outperform standard word lookup approach, these models have not been compared in any sort of systematic way, nor do we know how well they represent morphology (Table 2.11). To have a better understanding of these questions, Chapter 3 will present a systematic comparison of different subword units models and investigate how these models interact with morphological typology.

2.4 Dependency Parsing

Next, we discuss dependency parsing, which we will use to assess the character-level models in Chapters 4 and 5.

The main goal of dependency parsing is to automatically extract dependencies between words in a sentence in the form of graph structure. It is inspired by the **dependency grammar**, which states that syntactic structure is made up of binary, asymmetrical relations between lexical items called **dependencies**. In the dependency graph structure, **nodes** are words and each edge relates a syntactically subordinate word (the **dependent**) to another word on which it depends (the **head**). Each edge has a **dependency label** that defines the type of dependency. An *artificial* ROOT token is inserted at the beginning of the sentence and acts as the root of the dependency graph. Due to this single rooted structure, sometimes dependency graph is also called dependency tree. In general, any dependency graphs are expected to satisfy (1) the single-head constraint, which ensures that each node can only have exactly one head and (2) the acyclicity constraint, that enforces graphs to have no cycle. Figure 2.3 shows a dependency graph for 'She wrote me a letter'. In this example, we say that 'wrote' is the *head* of 'She' and that 'She' is the *dependent* of 'wrote'. Finally, the dependency label NSUBJ defines that 'She' is the *subject* of the predicate 'wrote'.

One important property of dependency graphs is **projectivity**. A dependency graph

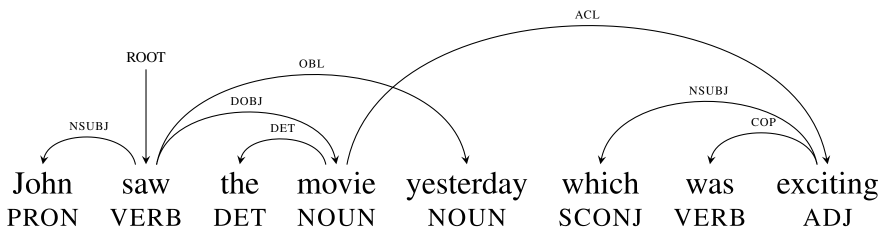


Figure 2.4: A *non-projective* dependency graph.

is said to be projective if, when we put the words in their linear order, preceded by an artificial ROOT, and draw dependencies between them in the semi-plane above the words, there exist no *crossing edges*. When a dependency graph has crossing edges, it is called **non-projective**. For example, dependency graph in Figure 2.3 is projective while dependency graph in Figure 2.4 is non-projective because the edge ('saw' $\rightarrow$ 'yesterday') and the edge ('movie' $\rightarrow$ 'exciting') are crossing. Non-projectivity mostly exists in languages with flexible word order or in complex sentences with long distance dependencies. Since many languages require non-projective dependency analysis, it is important to build parsers that can represent and parse non-projective dependencies (McDonald et al., 2005; McDonald and Satta, 2007).

The most common method for evaluating dependency parses are **Unlabeled Attachment Score** (UAS) and **Labeled Attachment Score** (LAS). UAS measures the percentage of words that are assigned the correct head, while LAS measures the percentage of words that are assigned the correct head and the correct dependency label.

Dependency analyses have been shown to be useful in many NLP applications, such as question answering (Cui et al., 2005), machine translation (Carreras and Collins, 2009), and information extraction (Angeli et al., 2015), among others. Dependency structure is often preferred to traditional lexical phrase structure due to its ability to explicitly encode predicate-argument structure. It is also independent of word order, making it more suitable for morphologically rich languages which have relatively free word order.

Parsing Morphologically Rich Languages The growing interest in using dependency analysis for multilingual NLP has led to the development of statistical parsing methods in the past two decades. In 2006 and 2007, the Conference on Computational Natural Language Learning (CoNLL) held shared tasks on multilingual dependency parsing (Buchholz and Marsi, 2006; Nivre et al., 2007) with the goal to uniformly evaluate different parsing methods on typologically diverse languages. Based on the shared task results on nine languages, the 2007 shared task organizers conclude that "the most difficult languages are those that combine a relatively free word order with a high degree of inflection" and hypothesize that "highly inflected languages with (relatively) free word order need more training data" (Nivre et al., 2007). In light of this finding, subsequent workshops and shared tasks on parsing morphologically rich languages (SPMRL; Seddah et al., 2013, 2014) were held to further advance studies on this topic. Tsarfaty et al. (2010) identify three main challenges in parsing morphologically rich languages, namely (1) the *architectural challenge* which concerns about the nature of input of the parsing system, (2) the *modeling challenge* which defines what type of morphological information to be included in the parsing model, and (3) the *lexical challenge* due to the high-level of morphological variation, that is how to guess morphological analyses of an OOV words.

Beside parsing models, there were also efforts in developing more treebanks for languages other than English. In particular, there was an initiative to develop cross-lingual treebanks with consistent syntactic dependency annotation across languages. This was started by the development of Stanford Dependencies (de Marneffe et al., 2006; de Marneffe and Manning, 2008), the Universal Dependency Treebank (UDT McDonald et al., 2013), and then the Universal Dependencies (UD; Nivre et al., 2016). At the time of the writing, the most recent UD version (2.4) covers more than 100 treebanks in over 70 languages, way more than what was available back in CoNLL 2006 and 2007 shared tasks. Indeed, the release of UD treebanks has prompted substantial research on multilingual and cross-lingual parsing, including the VarDial (Zampieri et al., 2017) and the CoNLL UD shared tasks on multilingual dependency parsing (Zeman et al., 2017a, 2018).

With the release of UD treebanks in multiple languages and the success of neural network based models, it is natural to ask whether the same morphological challenges identified by Tsarfaty et al. (2010) still exist when parsing morphologically rich languages. In particular, with the claim that character-level models learn morphology (Ling et al., 2015a; Lee et al., 2017), we wonder if neural parsers with character-level representations can solve the challenges (2) and (3). Next, we discuss previous studies which try to shed light on this question by applying character-level models for dependency parsing.

Character-Level Models for Dependency Parsing For neural dependency parsing, Ballesteros et al. (2015) is the first to use a character-level model in a transition-based parser. They employ a charLSTM model to encode words and obtain improvements over baseline word-level model on many morphologically rich languages. In the CoNLL 2017 UD shared task on multilingual dependency parsing, the top three best systems use different parsing models but all use character-level models to represent words (Dozat et al., 2017; Shi et al., 2017; Björkelund et al., 2017), showing that they are effective across many typologies. In the following 2018 shared task, almost all reported systems use character-level models in their systems. CharLSTM and charCNN are the two most popular choice of character-level models in the shared tasks (Zeman et al., 2017b, 2018).

On the interaction between character-level features, pretrained word embeddings, and POS tags, Smith et al. (2018b) observe that there are complex interactions between the three when they are used in a transition-based dependency parser. When used in isolation, each of them improves performance over a baseline system with randomly initialized word embeddings. However, combining them in any form—two at a time, or all three–often leads to a drop in performance. Their analysis shows that character-level models are more important for rare and open-class words, while POS tag information helps for disambiguating high-frequency function words. Stymne et al. (2018) use the concatenation of the word-level embedding, a character-level embedding obtained using charLSTM, and a treebank embedding as inputs for a multilingual model, and see improvements over the monolingual baseline. In a related work, de Lhoneux et al. (2018) study the parameter sharing of words, characters, and other model parameters for multilingual parsers. They find that the benefit of sharing word and character parameters is more dependent on the languages, suggesting that they are highly sensitive to phonological and morphosyntactic differences across languages. In line with these studies, we also examine why character-level models are effective for dependency parsing. In Chapter 4, we will study why character-level models are better than word-level baseline model but worse than *oracle* with access to explicit morphological analysis for monolingual parsers. We then extend this study to low-resource, multilingual parsing models in Chapter 5.

Chapter 3

Representing Morphology using Character-level Models

This chapter starts our work on understanding whether subword unit representation models learn morphology. To provide a strong baseline, we survey the fundamental experimental questions of subword unit models: the choice of subword unit, compositional model, and the interaction of morphological typology. We focus on language modeling—the standard task for learning word representations—and evaluate these models on ten typologically diverse languages. Our results extend previous findings that character-level models are effective across typologies, and this is because they learn orthographic similarities that are consistent with morphology. But we also find room for improvement: these strong baselines are still not as good as a model with access to morphology, even when learned from an order of magnitude more data. This chapter is based on ACL 2017 publication (Vania and Lopez, 2017).

3.1 Motivation

How should we encode morphology in the word representations? In Chapter 2, we briefly reviewed some of the word representation models, starting from the standard word-level models to models which exploit subword-level information such as morphological segments, character n-grams, or characters. Using subword units to represent morphology is appealing because it eliminates the needs of morphological annotations or analyzers which are expensive to built and hardly available for many languages. Subword unit representation models can seamlessly represent rare and OOV words and they also have fewer parameters to learn.

Does NLP still benefit from prior knowledge of morphology, or can they be replaced entirely by models of subword units? The relative merits of word, subword, and character-level representation models are not fully understood because each new model has been compared on different tasks and datasets, and often compared against word-level models. To understand whether morphology is captured by these models, we first need to build a solid subword-level baseline, by focusing on following questions:

1. How do representations based on subword units compare with each other?
2. What is the best way to compose subword representations?

To answer these questions, we performed a systematic comparison across different models for the simple and ubiquitous task of language modeling. We present experiments that vary (i) the type of subword unit; (ii) the composition function; and (iii) the morphological typology.

Chapters 1 and 2 discuss the fact that languages are typologically diverse with respect to their morphological processes, their degree of synthesis, and the types of morphological features that are encoded by these processes. Most of the subword unit models implicitly assume concatenative morphology, but many widely-spoken languages feature non-concatenative morphology, and it is unclear how such models will behave on these languages. Thus, it is also natural to ask:

3. Do character-level models capture morphology in terms of predictive accuracy?
4. How do different representations interact with languages of different morphological typologies?

To shed light on these questions, we further present oracle experiments by comparing our solid baselines against gold morphological analyses. Our results show that:

1. For most languages, character-level representations outperform the standard word representations. Most interestingly, a previously unstudied combination of character trigrams composed with biLSTMs performs best on the majority of languages.
2. BiLSTMs and CNNs are more effective composition functions than addition.
3. Character-level models learn functional relationships between orthographically

Unit	**Output of σ(*wants*)**
Morfessor	^want, s$
BPE	^w, ants$
char-trigram	^wa, wan, ant, nts ts$
character	^, w, a, n, t, s, $
analysis	want+VB, +3RD, +SG, +PRES

Table 3.1: Input representations for *wants*.

similar words, but don't (yet) match the predictive accuracy of models with access to true morphological analyses.

4. Character-level models are effective across a range of morphological typologies, but orthography influences their effectiveness.

3.2 Compositional Word Representation Models

We compare ten different models, varying subword units and composition functions that have commonly been used in recent work, but evaluated on various different tasks (see Table 2.11). Given a word w, we compute its representation $\mathbf{w}$ as:

$$\mathbf{w} = f(\mathbf{W}_s, \sigma(w)) \tag{3.1}$$

where σ is a deterministic function that returns a sequence of subword units; $\mathbf{W}_s$ is a parameter matrix of representations for the vocabulary of subword units; and f is a composition function which takes $\sigma(w)$ and $\mathbf{W}_s$ as input and returns $\mathbf{w}$. All of the representations that we consider take this form, varying only in f and σ.

3.2.1 Subword Units

We consider four variants of σ in Equation 3.1, each returning a different type of subword unit: character, character trigram, or one of two types of morph which are obtained from Morfessor 2.0 (Smit et al., 2014) or BPE segmentation. For Morfessor, we use default parameters while for BPE we set the number of merge operations to

10,000.[1,2] When we segment into character trigrams, we consider all trigrams in the word, including those covering notional beginning and end of word characters, as in Sperr et al. (2013). Example output of σ is shown in Table 3.1.

3.2.2 Composition Functions

Given a word $w = s_1, s_2, \ldots, s_n$, each subword s_i is represented by its embedding $\mathbf{s}_i$. We use three variants of f in Equation 3.1 to produce word representation $\mathbf{w}$:

- **Addition.** Our first composition function constructs the representation $\mathbf{w}$ by adding the representations of its subwords:

  $$\mathbf{w} = \sum_{i=1}^{n} \mathbf{s}_i \tag{3.2}$$

  The only subword unit that we don't compose by addition is characters, since this will produce the same representation for many different words ('poodle' vs. 'looped').

- **BiLSTM.** Our second composition function is a bidirectional long short term memory network (biLSTM), which we adapt based on its use in the character-level model of Ling et al. (2015a), as described in §2.3.2.

- **CNN.** The third composition function is a convolutional neural network (**CNN**) with highway layers, as in Kim et al. (2016) (§2.3.2). Since it can learn character n-grams directly, we only use the CNN with character input.

3.2.3 Language Model

We evaluate all the subword unit models on a language modeling task (§2.2). Our language model is an LSTM variant of RNN language model (§2.2). Let $s = w_1, \ldots, w_T$ be a sequence of words in a sentence. At each time step t, our LSTM receives input w_t

[1]BPE takes a single parameter: the number of merge operations. We tried different parameter values (1k, 10k, 100k) and manually examined the resulting segmentation on the English dataset. Qualitatively, 10k gave the most plausible segmentation and we used this setting across all languages.

[2]The best reported F-1 score for Morfessor 2.0 are: 0.749 for English, 0.558 for German, 0.432 for Turkish, 0.342 for Arabic, and 0.501 for Finnish (Bergmanis and Goldwater, 2017). These scores are calculated on the MorphoChallenge 2010 datasets: http://morpho.aalto.fi/events/morphochallenge2010.

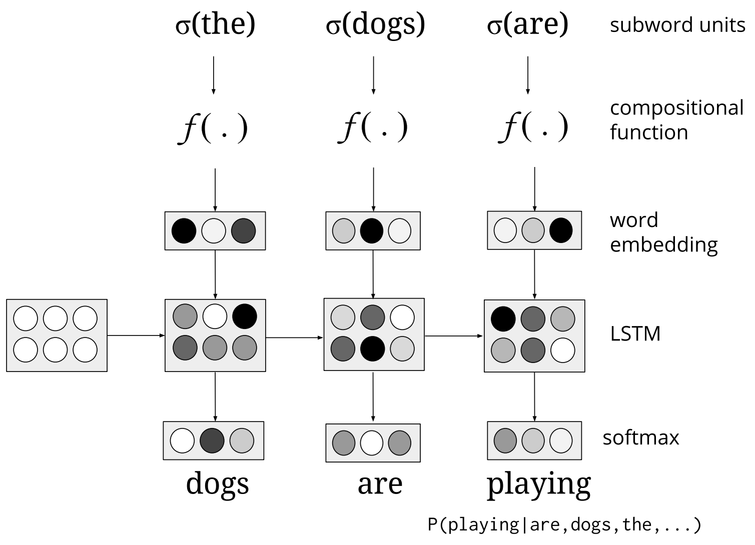


Figure 3.1: Our LSTM-LM architecture.

and predicts y_{t+1}. Using Equation 3.1, it first computes representation $\mathbf{w}_t$ of w_t. Given this representation and previous state $\mathbf{h}_{t-1}$, it produces a new state $\mathbf{h}_t$ and predicts y_{t+1}:

$$\mathbf{h}_t = \text{LSTM}(\mathbf{w}_t, \mathbf{h}_{t-1}) \tag{3.3}$$

$$\hat{\mathbf{y}}_{t+1} = \text{softmax}(\mathbf{V}^T \cdot \mathbf{h}_t) \tag{3.4}$$

where $\hat{\mathbf{y}}_{t+1}$ is the probability distributions over the words in the vocabulary. Note that this design means that we can *predict* only words from a finite output vocabulary, so our models differ only in their representation of context words. This design makes it possible to compare language models using perplexity, since they have the same event space.[3,4] The complete architecture of our system is shown in Figure 3.1, showing segmentation function σ and composition function f from Equation 3.1.

3.3 Experiments

We perform experiments on ten languages with varying dominant morphological processes (Table 3.2). We use datasets from Ling et al. (2015a) for English and Turk-

[3]Note that, while perplexities across event spaces are not directly comparable, we can still compare their log-likehood and there are tricks to convert them into perplexities (Cotterell et al., 2018; Mielke and Eisner, 2019).

[4]Open vocabulary word prediction which was a future work at the time our work is done is studied recently by Matthews et al. (2018).

Dominant Typology	Languages	#tokens	#types
Fusional	Czech	1.2M	125.4K
	English	1.2M	81.1K
	Russian	0.8M	103.5K
Agglutinative	Finnish	1.2M	188.4K
	Japanese	1.2M	59.2K
	Turkish	0.6M	126.2K
Root & Pattern	Arabic	1.4M	137.5K
	Hebrew	1.1M	104.9K
Reduplication	Indonesian	1.2M	76.5K
	Malaysian	1.2M	77.7K

Table 3.2: Statistics of our datasets.

ish. For Czech and Russian, we compile sentences from Universal Dependencies (UD) v1.3 (Nivre et al., 2015), taking the word forms (second column in CoNLL-U format), but skipping multiword tokens and empty nodes.[5] For other languages, we use preprocessed Wikipedia datasets from Al-Rfou et al. (2013) which already pre-tokenized.[6] The Arabic and Hebrew datasets are unvocalized, while the Japanese dataset mixes Kanji, Katakana, Hiragana, and Latin characters (for foreign words). Hence, a Japanese character can correspond to a character, syllable, or word. For each dataset, we use approximately 1.2M tokens to train, and approximately 150K tokens each for development and testing. Preprocessing involves lowercasing (except for character models) and removing hyperlinks.

To ensure that we compared models and not implementations, we reimplemented all models in a single framework using Tensorflow (Abadi et al., 2015).[7] We use a common setup for all experiments based on that of Ling et al. (2015a), Kim et al. (2016), and Miyamoto and Cho (2016). In preliminary experiments, we confirmed that our models produced similar patterns of perplexities for the reimplemented word and char-

[5]In UD, multiword tokens are indexed with integer ranges (1-2, 3-4, etc.) while empty nodes are defined with decimals (2.1, 2.2, etc.).

[6]Wikipedia datasets are available at https://sites.google.com/site/rmyeid/projects/polyglot

[7]Our implementation of these models can be found at https://github.com/claravania/subword-lstm-lm

acter LSTM models of Ling et al. (2015a). Even following detailed discussion with Ling (p.c.), we were unable to reproduce their perplexities exactly—our English reimplementation gives lower perplexities; our Turkish higher—but we do reproduce their general result that character bi-LSTMs outperform word models. We suspect that different preprocessing and the stochastic learning explains differences in perplexities. Our final model with bi-LSTMs composition follows Miyamoto and Cho (2016) as it gives us the same perplexity results for our preliminary experiments on the Penn Treebank dataset (Marcus et al., 1993), preprocessed by Mikolov et al. (2010).

3.3.1 Training and Evaluation

Our LSTM-LM uses two hidden layers with 200 hidden units and representation vectors for words, characters, and morphs all have dimension 200. All parameters are initialized uniformly at random from -0.1 to 0.1, trained by stochastic gradient descent with mini-batch size of 32, time steps of 20, for 50 epochs. To avoid overfitting, we apply dropout with probability 0.5 on the input-to-hidden layer and all of the LSTM cells (including those in the biLSTM, if used). For all models which do not use biLSTM composition, we start with a learning rate of 1.0 and decrease it by half if the validation perplexity does not decrease by 0.1 after 3 epochs. For models with biLSTMs composition, we use a constant learning rate of 0.2 and stop training when validation perplexity does not improve after 3 epochs. For the character CNN model, we use the same settings as the *small model* of Kim et al. (2016).

To make our results comparable to Ling et al. (2015a), for each language we limit the output vocabulary to the most frequent 5000 training words plus an unknown word token. To learn to predict unknown words, we follow Ling et al. (2015a): in training, words that occur only once are stochastically replaced with the unknown token with probability 0.5. To evaluate the models, we compute perplexity on the test data.

3.4 Results and Analysis

Table 3.3 presents our main results. In six of ten languages, character-trigram representations composed with biLSTMs achieve the lowest perplexities. As far as we know, this particular model has not been tested before, though character n-gram composed using addition has been proposed in the past (Sperr et al., 2013; Wieting et al., 2016; Bojanowski et al., 2017). We can see that the performance of character, character trigrams, and BPE are very competitive. Composition by biLSTMs or CNN is more effective than addition, except for Turkish. We also observe that BPE always outperforms Morfessor, even for the agglutinative languages. We now turn to a more detailed analysis by morphological typology.

Fusional languages. For these languages, character trigrams composed with biLSTMs outperformed all other models, particularly for Czech and Russian (up to 20%), which is unsurprising since both are morphologically richer than English.

Agglutinative languages. We observe different results for each language. For Finnish, character trigrams composed with biLSTMs achieves the best perplexity. Surprisingly, for Turkish character trigrams composed via addition is best and addition also performs quite well for other representations, potentially useful since the addition function is simpler and faster than biLSTMs. We suspect that this is due to the fact that Turkish morphemes are reasonably short, hence well-approximated by character trigrams. For Japanese, the improvements from character models are more modest than in other languages which we suspect due to the complexity of its orthography.

Root and Pattern. For these languages, character trigrams composed with biLSTMs also achieve the best perplexity. We had wondered whether CNNs would be more effective for root-and-pattern morphology, but since these data are unvocalized, it is more likely that non-concatenative effects are minimized, though we do still find morphological variants with consonantal inflections that behave more like concatenation. For example, *maktab* (root:*ktb*) is written as *mktb*. We suspect this makes character trigrams quite effective since they match the tri-consonantal root patterns among words which share the same root.

Reduplication. For Indonesian, BPE morphs composed with biLSTMs model obtain the best perplexity. For Malay, the character CNN outperforms other models. However, these improvements are small compared to other languages. This likely reflects that Indonesian and Malay are only moderately inflected, where inflection involves both concatenative and non-concatenative processes.

So far, we have compared subword unit representation models across different typologies. We answer Questions 1 and 2 in §3.1 by showing character-level models com-

Language	word	character		char trigrams		BPE		Morfessor		%imp
		biLSTM	CNN	add	biLSTM	add	biLSTM	add	biLSTM	
Czech	41.5	34.3	36.6	42.7	**33.6**	50.0	33.7	47.8	36.9	19.0
English	46.4	43.5	44.7	45.4	**43.0**	47.5	43.3	49.7	49.7	7.4
Russian	34.9	28.4	29.5	35.2	**27.7**	40.1	28.5	39.6	31.3	20.6
Finnish	24.2	20.1	20.3	24.9	**18.6**	26.8	19.1	27.8	22.5	23.1
Japanese	98.1	98.1	**91.6**	102.0	101.1	126.5	96.8	112.0	99.2	6.6
Turkish	67.0	54.5	55.1	**50.1**	54.2	59.5	57.3	62.2	62.7	25.2
Arabic	48.2	42.0	43.2	50.9	**39.9**	50.9	42.8	52.9	45.5	17.3
Hebrew	38.2	31.6	33.2	39.7	**30.4**	44.2	32.9	44.9	34.3	20.5
Indonesian	46.1	45.5	46.6	58.5	46.0	59.2	**43.4**	59.3	44.9	5.9
Malay	54.7	53.0	**50.6**	68.5	50.7	69.0	51.2	68.2	52.5	7.5

Table 3.3: Language model perplexities on test. The best model for each language is highlighted in **bold** and the improvement of this model over the word-level model is shown in the final column.

posed with biLSTM as strong baselines for representing morphology. The next set of analyses focuses on Questions 3 and 4; whether character-level models capture morphology in terms of predictive accuracy and their interactions with language typology.

Languages	addition	bi-LSTM	baseline
Czech	51.8	**30.07**	33.6
Russian	41.82	**26.44**	27.7

Table 3.4: Perplexity results using gold (hand-annotated) morphological analyses. **baseline** shows the best perplexities achieved by subword unit models (character trigrams composed with biLSTM (cf. Table 3.3)).

3.4.1 Effects of Morphological Analysis

Do character-level capture morphology in terms of predictive accuracy? In the experiments above, we used unsupervised morphological *segmentation* as a proxy for morphological *analysis* (Table 3.1). However, this is quite approximate, so it is natural to wonder what would happen if we had the true morphological analysis. If character-level models are powerful enough to capture the effects of morphology, then they should have the predictive accuracy of a model with access to this analysis. To find out, we conducted an oracle experiment using the human-annotated morphological analyses provided in the UD datasets for Czech and Russian, the only languages in our set for which these analyses were available. In these experiments we treat the lemma and each morphological feature as a subword unit.

The results (Table 3.4) show that biLSTM composition of these representations outperforms all other models for both languages. These results demonstrate that neither character representations nor unsupervised segmentation is a perfect replacement for manual morphological analysis, at least in terms of predictive accuracy. In light of character-level results, they imply that current unsupervised morphological analyzers are poor substitutes for real morphological analysis.

However, we can obtain much more unannotated than annotated data, and we might guess that the character-level models would outperform those based on morphological analyses if trained on larger data. To test this, we ran experiments that varied the training data size on three representation models: word, character-trigram biLSTM, and character CNN. Since we want to see how much training data is needed to reach perplexity obtained using annotated data, we use the same output vocabulary derived from the original training. While this makes it possible to compare perplexities across models, it is unfavorable to the models trained on larger data, which may focus on other words. This is a limitation of our experimental setup, but does allow us to draw

#tokens	word	char trigram bi-LSTM	char CNN
1M	39.69	32.34	35.15
2M	37.59	36.44	35.58
3M	36.71	35.60	35.75
4M	35.89	32.68	35.93
5M	35.20	34.80	37.02
10M	35.60	35.82	39.09
using morphological analysis			**30.07**

Table 3.5: Perplexity results on the Czech development data, varying training data size. Perplexity using ~1M tokens annotated data is **28.83**.

some tentative conclusions. As shown in Table 3.5, a character-level model trained on an order of magnitude more data still does not match the predictive accuracy of a model with access to morphological analysis.

3.4.2 Automatic Morphological Analysis

The oracle experiments show promising results if we have annotated data. But these annotations are expensive, so we also investigated the use of automatic morphological analysis. We obtained analyses for Arabic with the MADAMIRA 1.0 (Pasha et al., 2014). On the Penn Arabic Treebank corpus, MADAMIRA obtains 84.1% accuracy for predicting the correct morphological features (cf. EVALFULL metric in the original paper).[8] As in the experiment using annotations, we treated each morphological feature as a subword unit. The resulting perplexities of **71.94** and **42.85** for addition and bi-LSTMs, respectively, are worse than those obtained with character trigrams (**39.87**), though it approaches the best perplexities.

3.4.3 Targeted Perplexity Results

A difficulty in interpreting the results of Table 3.3 with respect to specific morphological processes is that perplexity is measured for all words. But these processes do not

[8]We only experimented with Arabic since MADAMIRA disambiguates words in contexts; most other analyzers we found did not do this, and would require additional work to add disambiguation.

Inflection	Model	all	frequent	rare
Czech	word	61.21	56.84	72.96
nouns	characters	51.01	<u>47.94</u>	59.01
	char-trigrams	<u>50.34</u>	48.05	<u>56.13</u>
	BPE	53.38	49.96	62.81
	morph. analysis	**40.86**	**40.08**	**42.64**
Czech	word	81.37	74.29	99.40
verbs	characters	70.75	68.07	77.11
	char-trigrams	<u>65.77</u>	<u>63.71</u>	<u>70.58</u>
	BPE	74.18	72.45	78.25
	morph. analysis	**59.48**	**58.56**	**61.78**
Russian	word	45.11	41.88	48.26
nouns	characters	37.90	37.52	<u>38.25</u>
	char-trigrams	<u>36.32</u>	<u>34.19</u>	38.40
	BPE	43.57	43.67	43.47
	morph. analysis	**31.38**	**31.30**	**31.50**
Russian	word	56.45	47.65	69.46
verbs	characters	45.00	40.86	50.60
	char-trigrams	<u>42.55</u>	<u>39.05</u>	<u>47.17</u>
	BPE	54.58	47.81	64.12
	morph. analysis	**41.31**	**39.8**	**43.18**

Table 3.6: Average perplexities of words that occur after nouns and verbs. Frequent words occur more than ten times in the training data; rare words occur fewer times than this. The best perplexity is in **bold** while the second best is <u>underlined</u>.

apply to all words, so it may be that the effects of specific morphological processes are washed out. To get a clearer picture, we measured perplexity for only specific subsets of words in our test data: specifically, given target word w_i, we measure perplexity of word w_{i+1}. In other words, we analyze the perplexities *when the inflected words of interest are in the most recent history*, exploiting the recency bias of our LSTM-LM. This is the perplexity most likely to be strongly affected by different representations, since we do not vary representations of the predicted word itself.

We look at several cases: nouns and verbs in Czech and Russian, where word classes can be identified from annotations, and reduplication in Indonesian, which we can

Language	type-level (%)	token-level (%)
Indonesian	1.10	2.60
Malay	1.29	2.89

Table 3.7: Percentage of full reduplication on the type and token level.

Model	all	frequent	rare
word	101.71	91.71	156.98
characters	**99.21**	**91.35**	**137.42**
BPE	117.2	108.86	156.81

Table 3.8: Average perplexities of words that occur after reduplicated words in the test set.

identify mostly automatically. For each analysis, we also distinguish between *frequent* cases, where the inflected word occurs more than ten times in the training data, and *rare* cases, where it occurs fewer than ten times. We compare only bi-LSTM models.

For Czech and Russian, we again use the UD annotation to identify words of interest. The results (Table 3.6), show that manual morphological analysis uniformly outperforms other subword models, with an especially strong effect for Czech nouns, suggesting that other models do not capture useful predictive properties of a morphological analysis. We do however note that character trigrams achieve low perplexities in most cases, similar to overall results (Table 3.3). We also observe that the subword models are more effective for rare words.

For Indonesian, we exploit the fact that the hyphen symbol '-' typically separates the first and second occurrence of a reduplicated morpheme, as in the examples of Chapter 2.1.2. We use the presence of word tokens containing hyphens to estimate the percentage of those exhibiting reduplication. As shown in Table 3.7, the numbers are quite low. Table 3.8 shows results for reduplication. In contrast with the overall results, the BPE biLSTM model has the worst perplexities, while character biLSTM has the best, suggesting that these models are more effective for reduplication.

Looking more closely at BPE segmentation of reduplicated words, we found that only 6 of 252 reduplicated words have a correct word segmentation, with the reduplicated morpheme often combining differently with the notional start-of-word or hyphen character. One the other hand BPE correctly learns 8 out of 9 Indonesian prefixes and 4 out of 7 Indonesian suffixes.[9] This analysis supports our intuition that the improvement from BPE might come from its modeling of concatenative morphology.

3.4.4 Qualitative Analysis

Table 3.9 presents nearest neighbors under cosine similarity for in-vocabulary, rare, and out-of-vocabulary (OOV) words.[10] For frequent words, standard word embeddings are clearly superior for lexical meaning. Character and morph representations tend to find words that are orthographically similar, suggesting that they are better at modeling dependent than root morphemes. The same pattern holds for rare and OOV words. We suspect that the subword models outperform words on language modeling because they exploit affixes to signal word class. We also noticed similar patterns in Japanese.

We analyze reduplication by querying reduplicated words to find their nearest neighbors using the BPE biLSTM model. If the model were sensitive to reduplication, we would expect to see morphological variants of the query word among its nearest neighbors. However, from Table 3.10, this is not so. With the partially reduplicated query *berlembah-lembah*, we do not find the lemma *lembah*.

3.5 Follow-up Work

We note some follow-up work inspired by this study which might shed more light on this topic. Gerz et al. (2018a,b) conducted a large scale study of LSTM language models on 50 typologically diverse languages. They compared character-level CNN model with standard word lookup approach and n-gram statistical language model. Different from our study, they trained a language model that operates on the full vocabulary setup, keeping all rare words in the modeled data. They further highlighted the main problem with fixed vocabulary language model, that it gives overly optimistic perplexity scores and that it can obscure the actual difficulty in modeling language, especially when the language has large number of rare words due to productive morphological process. Based on their results, they reported a strong correlation between perplexity

[9]We use Indonesian affixes listed in Larasati et al. (2011)

[10]https://radimrehurek.com/gensim/

Model	Frequent Words			Rare Words		OOV words	
	man	*including*	*relatively*	*unconditional*	*hydroplane*	*uploading*	*foodism*
word	person	like	extremely	nazi	molybdenum	-	-
	anyone	featuring	making	fairly	your	-	-
	children	include	very	joints	imperial	-	-
	men	includes	quite	supreme	intervene	-	-
BPE LSTM	ii	called	newly	unintentional	emphasize	upbeat	vigilantism
	hill	involve	never	ungenerous	heartbeat	uprising	pyrethrum
	text	like	essentially	unanimous	hybridized	handling	pausanias
	netherlands	creating	least	unpalatable	unplatable	hand-colored	footway
char-trigrams LSTM	mak	include	resolutely	unconstitutional	selenocysteine	drifted	tuaregs
	vill	includes	regeneratively	constitutional	guerrillas	affected	quft
	cow	undermining	reproductively	unimolecular	scrofula	conflicted	subjectivism
	maga	under	commonly	medicinal	seleucia	convicted	tune-up
char-LSTM	mayr	inclusion	relates	undamaged	hydrolyzed	musagète	formulas
	many	insularity	replicate	unmyelinated	hydraulics	mutualism	formally
	mary	includes	relativity	unconditionally	hysterotomy	mutualists	fecal
	may	include	gravestones	uncoordinated	hydraulic	meursault	foreland
char-CNN	mtn	include	legislatively	unconventional	hydroxyproline	unloading	fordism
	mann	includes	lovely	unintentional	hydrate	loading	dadaism
	jan	excluding	creatively	unconstitutional	hydrangea	upgrading	popism
	nun	included	negatively	untraditional	hyena	upholding	endemism

Table 3.9: Nearest neighbours of semantically and syntactically similar words.

Query	Top nearest neighbours
kota-kota (*cities*)	wilayah-wilayah (*areas*), pulau-pulau (*islands*), negara-negara (*countries*), bahasa-bahasa (*languages*), koloni-koloni (*colonies*)
berlembah-lembah (*have many valleys*)	berargumentasi (*argue*), bercakap-cakap (*converse*), berkemauan (*will*), berimplikasi (*imply*), berketebalan (*have a thickness*)

Table 3.10: Nearest neighbours of Indonesian reduplicated words in the BPE bi-LSTM model.

and morphological typology; perplexity increases as the morphology becomes more complex, from isolating, fusional, to the agglutinative ones. However, their study does not yet address the question about *oracle morphology*, which would be an interesting extension of our study.

Godin et al. (2018) investigated whether neural networks trained for type-level morphological tagging task can discover patterns that coincides with morphological segmentations and annotations defined by humans ('-a' in 'económicas' indicates feminine gender). Using either CNN or biLSTM as word encoder, they measured the contributions of character sets by decomposing the output of neural networks into two distinct groups of contributions: (1) those originating from a specific character or character sets, and (2) those originating from all other characters within the same word. Experiments on three languages—Finish, Spanish, and Swedish—show that in almost all cases the attribution follows the manually defined segmentation/annotations. Further, they found the CNN based model keeps track on the most important suffix while biLSTM tend to focus on both root and suffix.

Another way to examine how neural models encode linguistic knowledge is by looking at their individual hidden units. Kementchedjhieva and Lopez (2018) examined individual hidden units in a character-level language model by hand and concluded that a character-level model can indeed "learn to identify linguistic units of higher order, such as morphemes and words" and also "learn some underlying linguistic properties and regularities of said units". Pinter et al. (2019) also examined hidden units of POS tagging models trained with character-level inputs but with automatic evaluation metric. They found that the ability character-level biLSTM model in identifying POS can be related to the typology properties of the language.

Finally, we also note some works that compare character-level models for other downstream tasks, including semantic role labeling (Sahin and Steedman, 2018a), machine translation (Ataman and Federico, 2018), and sentence pair modeling Lan and Xu (2018).

3.6 Conclusions

In this chapter, we systematically compared word representation models with different levels of morphological awareness, across languages with different morphological typologies. We confirm previous findings that character-level models are effective especially for languages with complex morphological processes. Our qualitative analysis suggests that they learn orthographic similarity of affixes, and lose the meaning of root morphemes. We conclude that character-level models are effective because they capture orthographic regularities that are consistent with morphology.

Across languages with different typologies, our experiments show that character-level models are most effective for agglutinative languages, followed by fusional languages, and less for analytical languages. However, we note that these results do not generalize to all languages, since factors such as morphology and orthography affect the utility of these representations.

To test whether character-level models capture morphology, we compare them against an oracle with access to gold morphological analysis. We show that character-level models do not match the predictive accuracy of our oracle, even when learned with an order of magnitude more data. This result suggests that character-level models do not entirely capture morphology. Which aspects of morphology are not captured by character-level models? In the next chapter, we will investigate this question more deeply by diagnosing the representation learned by both character-level models and the oracle.

Chapter 4

Character-Level Models and Morphology in Dependency Parsing

In the previous chapter, we have established character-level models as strong baselines for morphologically-aware word representations. Our analysis suggests that they learn orthographic similarities which are consistent with morphology and hence useful for representing rare and OOV words. However, our oracle experiments also indicate that character-level models do not entirely capture morphology. In this chapter, we ask, which aspects of morphology are difficult to learn by these models? To answer this question, we diagnose the character-level models and oracle performance on dependency parsing—a task that benefits substantially from morphology. We show that character-level models are poor in disambiguating words, particularly in the face of *case syncretism*. We then demonstrate that explicitly modeling morphological case improves our best parsing model, showing that character-level models can benefit from targeted forms of explicit morphological modeling. This chapter is based on the EMNLP 2018 publication (Vania et al., 2018) and its extension at the BlackBoxNLP workshop (Vania and Lopez, 2018). Research was conducted in collaboration with Andreas Grivas, which helped in the implementation of the dependency parser used in this study.

4.1 Motivation

The effectiveness of character-level models has raised a question and indeed debate about explicit modeling of morphology in NLP. Ling et al. (2015b) propose that "prior information regarding morphology ... among others, should be incorporated" into character-level models, while Chung et al. (2016) counter that it is "unnecessary to consider these prior information" when modeling characters. Whether we need to explicitly model morphology is a question whose answer has a real cost: as Ballesteros et al. (2015) note, morphological annotation is expensive, and this expense could be reinvested elsewhere if the predictive aspects of morphology are learnable from strings.

Morphemes typically have similar orthographic representations across words. Since character-level models produce similar representations for orthographically similar words, their effectiveness is often attributed to their ability in modeling morphology. In Chapter 1, we mentioned a claim from Ling et al. (2015a) that character-level models can "model the complex form-function relationship, which captures non-compositional effects, where small orthographic differences may correspond to large semantic or syntactic differences (*butter* vs. *batter*) ... in addition to the more regular effects due to, e.g., morphological processes.". In Chapter 3, we showed that character-level models learn orthographic similarity of affixes, but they tend to lose the meaning of root morphemes.

Do character-level models learn morphology? We view this as an empirical claim requiring empirical evidence. The claim has been tested implicitly by comparing character-level models to word lookup models on various NLP tasks (Kim et al., 2016; Ling et al., 2015a; Ballesteros et al., 2015; Belinkov et al., 2017). In this chapter, we test it explicitly, asking how character-level models compare with an oracle model with access to morphological annotations. This extends our oracle experiments in Chapter 3 showing that character-aware language models in Czech and Russian benefit substantially from oracle morphology. However, here we focus on dependency parsing (§2.4)—a task which benefits substantially from morphological knowledge—and we experiment with twelve languages using a variety of techniques to probe our models.

Our summary finding is that character-level models lag the oracle in nearly all languages (§4.2). The difference is small, but suggests that there is value in modeling morphology. When we tease apart the results by part of speech and dependency type, we trace the difference back to the character-level model's inability to disambiguate words even when encoded with arbitrary context (§4.3). Specifically, it struggles with *case syncretism*, in which noun case—and thus syntactic function—is ambiguous. We show that the oracle relies on morphological case, and that a character-level model provided *only* with morphological case rivals the oracle, even when case is provided by another predictive model (§4.4). Finally, we show that the crucial morphological features vary by language (§4.5).

4.1.1 Parsing Model

We use a neural graph-based dependency parser combining elements of two recent models (Kiperwasser and Goldberg, 2016; Zhang et al., 2017).[1] Let $w = w_1, \ldots, w_{|w|}$ be an input sentence of length $|w|$ and let w_0 denote an artificial ROOT token. We represent the ith input token w_i by concatenating its *word representation* (described in §4.1.2), $\mathbf{e}(w_i)$ and part-of-speech (POS) representation, $\mathbf{p}_i$:[2]

$$\mathbf{x}_i = [\mathbf{e}(w_i); \mathbf{p}_i] \tag{4.1}$$

We call $\mathbf{x}_i$ the *embedding* of w_i since it depends on context-independent word and POS representations. We obtain a context-sensitive *encoding* $\mathbf{h}_i$ with a bidirectional LSTM (biLSTM), which concatenates the hidden states of a forward and backward LSTM at position i. Using $\mathbf{h}_i^f$ and $\mathbf{h}_i^b$ respectively to denote these hidden states, we have:

$$\mathbf{h}_i = [\mathbf{h}_i^f; \mathbf{h}_i^b] \tag{4.2}$$

We use $\mathbf{h}_i$ as the final input representation of w_i.

4.1.1.1 Head prediction

For each word w_i, we compute a distribution over all other word positions $j \in \{0, ..., |w|\}/i$ denoting the probability that w_j is the headword of w_i.

$$P_{head}(w_j \mid w_i, w) = \frac{\exp(a(\mathbf{h}_i, \mathbf{h}_j))}{\sum_{j'=0}^{|w|} \exp(a(\mathbf{h}_i, \mathbf{h}_{j'}))} \tag{4.3}$$

[1]When this study was conducted, the model of Kiperwasser and Goldberg (2016) is the state-of-the-art for dependency parsing on English and Chinese.

[2]This combination yields the best labeled accuracy according to Ballesteros et al. (2015).

Here, a is a neural network that computes an association between w_i and w_j using model parameters $\mathbf{U}_a, \mathbf{W}_a$, and $\mathbf{v}_a$.

$$a(\mathbf{h}_i, \mathbf{h}_j) = \mathbf{v}_a \tanh(\mathbf{U}_a \mathbf{h}_i + \mathbf{W}_a \mathbf{h}_j) \tag{4.4}$$

4.1.1.2 Label prediction

Given a head prediction for word w_i, we predict its syntactic label $\ell_k \in L$ using a similar network.

$$P_{label}(\ell_k \mid w_i, w_j, w) = \frac{\exp(f(\mathbf{h}_i, \mathbf{h}_j)[k])}{\sum_{k'=1}^{|L|} \exp(f(\mathbf{h}_i, \mathbf{h}_j)[k'])} \tag{4.5}$$

where L is the set of output labels and f is a function that computes label score using model parameters $\mathbf{U}_\ell, \mathbf{W}_\ell$, and $\mathbf{V}_\ell$:

$$f(\mathbf{h}_i, \mathbf{h}_j) = \mathbf{V}_\ell \tanh(\mathbf{U}_\ell \mathbf{h}_i + \mathbf{W}_\ell \mathbf{h}_j) \tag{4.6}$$

The model is trained to minimize the summed cross-entropy losses of both head and label predictions. At test time, we apply the Chu-Liu-Edmonds (Chu and Liu, 1965; Edmonds, 1967) algorithm to ensure well-formed, possibly non-projective trees.

4.1.2 Word representations

To confirm our finding in Chapter 3 on subword unit models, in particular the character-level models, we again compare the word representation models of our neural parsers. We consider the following five models to compute the word representation $\mathbf{e}(w_i)$ in Equation 4.1:

- **word**. Every word type has its own learned vector representation.
- **charLSTM**. Characters are composed using a biLSTM (Ling et al., 2015a), and the final states of the forward and backward LSTMs are concatenated to yield the word representation. This model is equivalent to the charLSTM model described in §2.3.2.
- **charCNN**. Characters are composed using a convolutional neural network (Kim et al., 2016). This model is equivalent to the charCNN model described in §2.3.2.

- **trigramLSTM**. Character trigrams are composed using a biLSTM, an approach that we previously found to be effective across typologies (Vania and Lopez, 2017).
- **oracle**. We treat the morphemes of a morphological annotation as a sequence and compose them using a biLSTM. We only use universal inflectional features defined in the UD annotation guidelines (Appendix A.1). For example, the morphological annotation of *chases* is [CHASE, PERSON=3RD, NUM-SG, TENSE=PRES].

Throughout this chapter, we use the name of model as shorthand for the dependency parser that uses that model as input (Equation 4.1).

4.2 Experiments

Data We experiment on twelve languages with varying morphological typologies (Table 4.1) in the Universal Dependencies (UD) treebanks version 2.0 (Nivre et al., 2017).[3] Note that while Arabic and Hebrew follow a *root & pattern* typology, their datasets are unvocalized, which might reduce the observed effects of this typology. However, their UD annotations use a *two-level indexing scheme* such that the basic units of the dependency trees are morphemes instead of standard word forms.[4] Since morphological and syntactic annotation is only defined at the word form level, we ignore multiword tokens and empty nodes (for the analysis of ellipsis) in the treebanks, and only use the word form level information in all of our experiments.[5] We use gold sentence segmentation, tokenization, universal POS (UPOS), and morphological (XFEATS) annotations provided in UD.

Implementation and training Our Chainer (Tokui et al., 2015) implementation encodes words (Equation 4.2) in two-layer biLSTMs with 200 hidden units, and uses 100 hidden units for head and label predictions (output of Equations 4.4 and 4.6). We set batch size to 16 for charCNN and 32 for other models following a grid search. We

[3]For Russian we use the UD_Russian_SynTagRus treebank, and for all other languages we use the default treebank.

[4]These basic units are called syntactic words in the UD terminology.

[5]In UD, multiword tokens are indexed with integer ranges (1-2, 3-4, etc.) while empty nodes are defined with decimals (2.1, 2.2, etc.).

Languages	#sents (K)	#tokens (K)	type/token ratio (%)
Finnish	12.2	162.6	28.5
Turkish	3.7	38.1	33.6
Czech	68.5	1173.3	9.5
English	12.5	204.6	8.1
German	14.1	269.6	17.7
Hindi	13.3	281.1	6
Portuguese	8.3	206.7	11.7
Russian	48.8	870	11.4
Spanish	14.2	382.4	11.1
Urdu	4.0	108.7	8.8
Arabic	6.1	223.9	10.3
Hebrew	5.2	137.7	11.7

Table 4.1: Training data statistics. Languages are grouped by their dominant morphological processes, from top to bottom: agglutinative, fusional, and root & pattern.

apply dropout to the embeddings (Equation 4.1) and the input of the head prediction. We use Adam optimizer with initial learning rate 0.001 and clip gradients to 5, and train all models for 50 epochs with early stopping. For the word model, we limit our vocabulary to the 20K most frequent words, replacing less frequent words with an unknown word token. The charLSTM, trigramLSTM, and oracle models use a one-layer biLSTM with 200 hidden units to compose subwords. For charCNN, we use the small model setup of Kim et al. (2016).

Parsing Results Table 4.2 presents test results for every model on every language, establishing three results. First, they support previous findings that character-level models outperform word-based models—indeed, the charLSTM model outperforms the word model on LAS for all languages except Hindi and Urdu for which the results are identical.[6] Second, they establish strong baselines for the character-level models: the charLSTM generally obtains the best parsing accuracy, closely followed by charCNN. Third, they demonstrate that character-level models rarely match the accuracy

[6]Note that Hindi and Urdu are mutually intelligible.

Model →	word		charLSTM		charCNN		trigramLSTM		oracle		o/c
↓ Language	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	LAS
Finnish	85.7	80.8	**90.6**	**88.4**	89.9	87.5	89.7	87.0	90.6	88.8	+0.4
Turkish	71.4	61.6	**74.7**	**68.6**	74.4	67.9	73.2	65.9	75.3	69.5	+0.9
Czech	92.6	89.3	**93.5**	**90.6**	**93.5**	**90.6**	92.7	89.2	94.3	92.0	+1.4
English	90.6	88.9	91.3	89.4	**91.7**	**90.0**	90.4	88.5	91.7	89.9	+0.5
German	**88.1**	**84.5**	88.0	**84.5**	87.8	84.4	87.1	83.5	88.8	86.5	+2.0
Hindi	**95.8**	93.1	95.7	**93.3**	95.7	93.2	93.4	89.8	95.9	93.3	-
Portuguese	87.4	85.5	**87.8**	**86.0**	87.7	**86.0**	86.7	84.8	88.0	86.5	+0.5
Russian	92.4	90.1	**94.0**	**92.4**	93.8	92.1	92.0	89.5	94.4	93.3	+0.9
Spanish	89.4	86.9	89.8	**87.4**	**90.0**	87.3	88.6	85.5	90.0	87.7	+0.3
Urdu	91.1	87.0	91.2	87.1	**91.3**	**87.2**	88.6	83.5	90.9	87.0	-0.1
Arabic	75.5	70.9	**76.7**	72.1	76.6	**72.2**	74.6	68.9	76.7	72.7	+0.6
Hebrew	**73.5**	**69.8**	73.4	**69.8**	73.3	**69.8**	71.3	67.1	73.3	70.0	+0.2

Table 4.2: Unlabeled Attachment Score (UAS) and Labeled Attachment Score (LAS) on **test set**. The best accuracy for each language is highlighted in **bold** for all models, and for all non-oracle models. **o/c:** LAS improvement from charLSTM to oracle.

of an oracle model with access to explicit morphology. This reinforces our finding in Chapter 3: character-level models are effective tools, but they do not learn everything about morphology, and they seem to be closer to oracle accuracy in agglutinative rather than in fusional languages (Vania and Lopez, 2017).

4.3 Analysis

4.3.1 Why do characters beat words?

In character-level models, orthographically similar words share many parameters, so we would expect these models to produce good representations of OOV words that are morphological variants of training words. Does this effect explain why they are better than word-level models?

Language	dev	LAS improvement	
	%OOV	non-OOV	OOV
Finnish	23.0	6.8	17.5
Turkish	24.0	4.6	13.5
Czech	5.8	1.4	3.9
English	6.8	0.7	5.2
German	9.7	0.9	0.7
Hindi	4.3	0.2	0.0
Portuguese	8.1	0.3	1.3
Russian	8.4	2.1	6.9
Spanish	7.0	0.4	0.7
Arabic	8.0	1.2	7.3
Hebrew	9.0	0.2	1.3

Table 4.3: LAS improvements (charLSTM − word) for non-OOV and OOV words on development set.

4.3.1.1 Sharing parameters helps with both seen and unseen words

Table 4.3 shows how the character model improves over the word model for both non-OOV and OOV words. On the agglutinative languages Finnish and Turkish, where the OOV rates are 23% and 24% respectively, we see the highest LAS improvements, and we see especially large improvements in accuracy of OOV words. However, the effects are more mixed in other languages, even with relatively high OOV rates. In particular, languages with rich morphology like Czech, Russian, and (unvocalised) Arabic see more improvement than languages with moderately rich morphology and high OOV rates like Portuguese or Spanish. This pattern suggests that parameter sharing between pairs of *observed* training words can also improve parsing performance. For example, if “dog” and “dogs” are observed in the training data, they will share activations in their context *and* on their common prefix.

Language	Model	ADJ	NOUN	PRON	PROPN	VERB	Overall
Finnish	%tokens	8.1	32.5	8.2	6.7	16.1	-
	charLSTM	89.2	82.1	88.1	84.5	88.4	87.7
	oracle	90.3	83.3	89.5	86.2	89.3	88.5
	diff	+1.1	+1.2	+1.4	+1.7	+0.9	+0.8
Czech	%tokens	14.9	28.7	3.6	6.3	10.7	-
	charLSTM	94.2	83.6	85.3	84.3	90.7	91.2
	oracle	94.8	87.5	88.5	86.8	91.1	92.5
	diff	+0.6	+3.9	+3.2	+2.5	+0.4	+1.3
German	%tokens	7.6	20.4	9.5	5.6	12.1	-
	charLSTM	88.4	81.4	86.0	82.4	85.2	87.5
	oracle	89.1	87.1	93.2	84.4	86.3	89.7
	diff	+0.7	+5.7	+7.2	+2.0	+1.1	+2.2
Russian	%tokens	12.2	29.3	6.1	4.6	13.7	-
	charLSTM	93.2	86.7	92.0	80.2	88.5	91.6
	oracle	93.7	88.8	93.3	86.4	88.9	92.6
	diff	+0.5	+2.1	+1.3	+6.2	+0.4	+1.0

Table 4.4: Labeled accuracy for different parts of speech on development set.

4.3.2 Why do morphemes beat characters?

Let's turn to our main question: what do character-level models learn about morphology? To answer it, we compare the oracle model to charLSTM, our best character-level model.

4.3.2.1 Morphological analysis disambiguates words

In the oracle, morphological annotations disambiguate some words that the charLSTM must disambiguate from context. Consider these Russian sentences from Baerman et al. (2005):

(9) Maša čitaet pis′mo
Masha reads letter
'*Masha reads a letter.*'

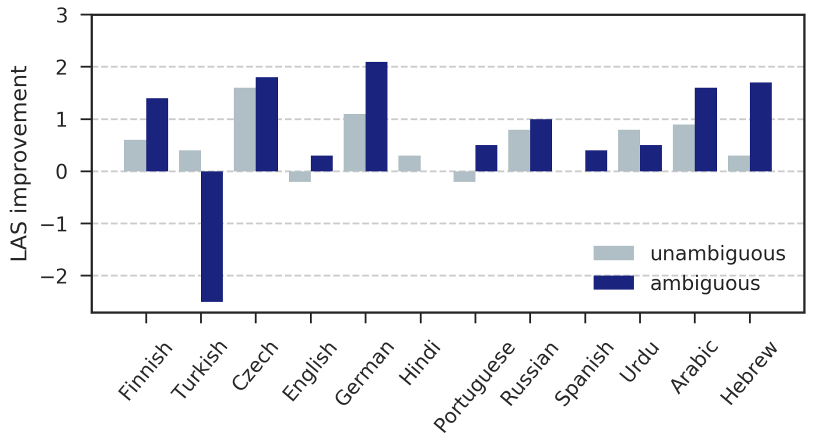


Figure 4.1: LAS improvements (oracle − charLSTM) for ambiguous and unambiguous words on development set.

(10) Na stole ležit <u>pis′mo</u>
on table lies <u>letter</u>
'*There's a <u>letter</u> on the table.*'

Pis′mo ("letter") acts as the subject in Example (9), and as object in Example (10). This knowledge is available to the oracle via morphological case: in Example (9), the case of *pis′mo* is nominative and in Example (10) it is accusative. Could this explain why the oracle outperforms the character model?

To test this, we look at accuracy for word types that are *empirically* ambiguous—those that have more than one morphological analysis in the training data. Note that by this definition, some ambiguous words will be seen as unambiguous, since they were seen with only one analysis. To make the comparison as fair as possible, we consider only words that were observed in the training data. Figure 4.1 compares the improvement of the oracle on ambiguous and seen unambiguous words, and as expected we find that handling of ambiguous words improves with the oracle in almost all languages. The only exception is Turkish, which has the least training data.

4.3.2.2 Morphology helps for nouns

Now we turn to a more fine-grained analysis conditioned on the annotated part-of-speech (POS) of the *dependent*. We focus on four languages where the oracle strongly outperforms the best character-level model on the development set: Finnish, Czech,

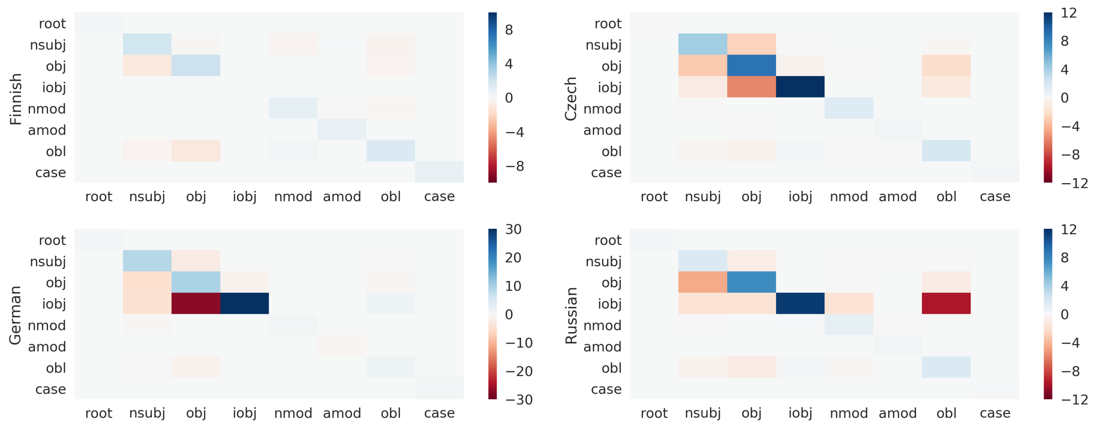


Figure 4.2: Heatmaps of the difference between oracle vs. charLSTM confusion matrices for label prediction when both head predictions are correct (**x-axis**: predicted labels; **y-axis**: gold labels). Blue cells have higher oracle values, red cells have higher charLSTM values.

German, and Russian.[7] We consider five POS categories that are frequent in all languages and consistently annotated for morphology in our data: adjective (ADJ), noun (NOUN), pronoun (PRON), proper noun (PROPN), and verb (VERB).

Table 4.4 shows that the three noun categories—ADJ, PRON, and PROPN—benefit substantially from oracle morphology, especially for the three fusional languages: Czech, German, and Russian.

4.3.2.3 Morphology helps for subjects and objects

We analyze results by the dependency type of the *dependent*, focusing on types that interact with morphology: *root*, nominal subjects (*nsubj*), objects (*obj*), indirect objects (*iobj*), nominal modifiers (*nmod*), adjectival modifier (*amod*), obliques (*obl*), and (syntactic) case markings (*case*).

Figure 4.2 shows the differences in the confusion matrices of the charLSTM and oracle for those words on which both models correctly predict the head. The differences on Finnish are small, which we expect from the similar overall LAS of both models. But for the fusional languages, a pattern emerges: the charLSTM consistently underper-

[7]This is slightly different than on the test set, where the effect was stronger in Turkish than in Finnish. In general, we found it difficult to draw conclusions from Turkish, possibly due to the small size of the data.

forms the oracle on nominal subject, object, and indirect object dependencies—labels closely associated with noun categories. From inspection, it appears to frequently mislabel objects as nominal subjects when the dependent noun is morphologically ambiguous. For example, in the sentence of Figure 4.3, *Gelände* ("terrain") is an object, but the charLSTM incorrectly predicts that it is a nominal subject. In the training data, *Gelände* is ambiguous: it can be accusative, nominative, or dative.

In German, the charLSTM frequently confuses objects and indirect objects. By inspection, we found 21 mislabeled cases, where 20 of them would likely be correct if the model had access to morphological case (usually dative). In Czech and Russian, the results are more varied: indirect objects are frequently mislabeled as objects, obliques, nominal modifiers, and nominal subjects. We note that indirect objects are relatively rare in these data, which may partly explain their frequent mislabeling.

4.4 Characters and case syncretism

So far, we've seen that for our three fusional languages—German, Czech, and Russian—the oracle strongly outperforms a character model on nouns with ambiguous morphological analyses, particularly on core dependencies: nominal subjects, objects and indirect objects. Since the nominative, accusative, and dative morphological cases are strongly (though not perfectly) correlated with these dependencies, it is easy to see why the morphologically-aware oracle is able to predict them so well. We hypothesized that these cases are more challenging for the character model because these languages feature a high degree of *syncretism*—functionally distinct words that have the same form—and in particular case syncretism. For example, referring back to examples (9) and (10), the character model must disambiguate *pis´mo* from its context, whereas the oracle can directly disambiguate it from a feature of the word itself.[8]

To understand this, we first designed an experiment to see whether the charLSTM could successfully disambiguate *noun* case, using a diagnostic classifier (Veldhoen et al., 2016a; Shi et al., 2016; Adi et al., 2017)—a method that has been used previously to analyze representations learned by neural machine translation models (Belinkov et al., 2017). We train a neural classifier that takes as input a word representation

[8]We are far from first to observe that morphological case is important to parsing: Seeker and Kuhn (2013) observe the same for non-neural parsers.

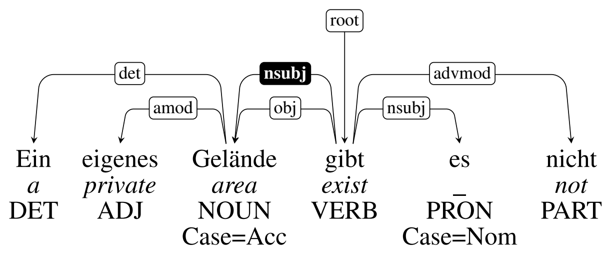


Figure 4.3: A sentence which the oracle parses perfectly (shown in white) and the charLSTM predicts an incorrect label (shown in black).

from the trained parser and predicts a morphological feature of that word—for example that its case is nominative (CASE=NOM). The classifier is a feedforward neural network with one hidden layer, followed by a ReLU non-linearity. We consider two representations of each word: its embedding ($\mathbf{x}_i$; Eq. 4.1) which is *context-insensitive*, and its encoding ($\mathbf{h}_i$; Eq 4.2) which is *context-sensitive*. To understand the importance of case, we consider it alongside number and gender features as well as whole feature bundles.

4.4.1 The oracle relies on case

Table 4.5 shows the results of morphological feature classification. The oracle embeddings have almost perfect accuracy—and this is just what we expect, since the representation only needs to preserve information from its input. For the fusional languages—Czech, German, and Russian—the charLSTM embeddings perform well on some features, except on case. This results suggest that the character-level models still struggle to learn case when given only the input text. We observe a slightly different pattern on Finnish results. The character embeddings achieves almost similar performance as the oracle embeddings. This results highlights the differences in morphological process between Finnish and the other fusional languages.

The classification results from the encoding are particularly interesting: the oracle still performs very well on morphological case, but less well on other features, even though they appear in the input. In the character model, the accuracy in morphological prediction also degrades in the encoding—*except* for case, where accuracy on case improves by up to ~19%.

These results make intuitive sense: representations learn to preserve information from

Language	Feature	baseline	embedding		encoder	
			char	oracle	char	oracle
Czech	Case	71.1	74.4	100	86.5	98.6
	Gender	92.9	98.1	100	71.2	58.6
	Number	88.9	94.7	100	84.2	84.8
	All	70.4	72.5	99.9	58.1	50.2
German	Case	35.2	35.7	100	80.8	99.7
	Gender	56.8	63.6	100	75.7	78.0
	Number	59.1	67.1	100	78.3	93.9
	All	34.0	34.3	100	63.6	78.5
Russian	Case	71.6	80.5	100	90.4	98.5
	Gender	87.7	97.4	100	69.9	57.3
	Number	83.7	94.5	100	85.7	83.8
	All	71.3	77.2	99.9	56.9	47.2
Finnish	Case	56.0	96.7	100	88.9	91.4
	Number	56.4	97.4	100	81.9	89.5
	All	55.8	95.0	91.6	74.0	82.7

Table 4.5: Morphological tagging accuracy from representations using the charLSTM and oracle embedding and encoder representations. Baseline simply chooses the most frequent tag. *All* means we concatenate all annotated features in UD as one tag.

their input that is useful for subsequent predictions. In our parsing model, morphological case is very useful for predicting dependency labels, and since it is present in the oracle's input, it is passed almost completely intact through each representation layer. The character model, which must disambiguate case from context, draws as much additional information as it can from surrounding words through the LSTM encoder. But other features, and particularly whole feature bundles, are presumably less useful for parsing, so neither model preserves them with the same fidelity.[9]

[9]This finding is consistent with Ballesteros (2013) which performed careful feature analysis on morphologically rich languages and found that lemma and case features provide the highest improvement in a non-neural transition based parser compared to other features.

4.4.2 Explicitly modeling case improves parsing accuracy

Our analysis indicates that case is important for parsing, so it is natural to ask: Can we improve the neural model by explicitly modeling case? To answer this question, we ran a set of experiments, considering two ways to augment the charLSTM with case information: multitask learning (MTL; Caruana, 1997) and a pipeline model in which we augment the charLSTM model with either predicted or gold case. For example, we use [p, i, z, z, a, NOM] to represent *pizza* with nominative case. For MTL, we follow the setup of Søgaard and Goldberg (2016) and Coavoux and Crabbé (2017). We increase the biLSTMs layers from two to four and use the first two layers to predict morphological case, leaving out the other two layers specific only for parser.

For the pipeline model, we first train a morphological tagger to predict morphological case. We adapt the parser's encoder architecture for our morphological tagger. Following notation in Section 4.1.1, each word w_i is represented by its context-sensitive encoding, $\mathbf{h}_i$ (Eq. 4.2). The encodings are then fed into a feed-forward neural network with two hidden layers—each has a ReLU non-linearity—and an output layer mapping the to the morphological tags, followed by a softmax. We set the size of the hidden layer to 100 and use dropout probability 0.2. We use Adam optimizer with initial learning rate 0.001 and clip gradients to 5. We train each model for 20 epochs with early stopping. The model is trained to minimized the cross-entropy loss.

Since we do not have additional data with the same annotations, we use the same UD dataset to train our tagger. To prevent overfitting, we only use the first 75% of training data for training.[10] After training the taggers, we predict the case for the training, development, and test sets and use them for dependency parsing. This tagger does not share parameters with the parser.

Table 4.6 summarizes the results on Czech, German, and Russian. We find augmenting the charLSTM model with either oracle or predicted case improve its accuracy, although the effect is different across languages. The improvements from predicted case results are interesting, since in non-neural parsers, predicted case usually harms accuracy (Tsarfaty et al., 2010). However, we note that our taggers use gold POS, which might help. The MTL models achieve similar or slightly better performance than the character-only models, suggesting that supplying case in this way is benefi-

[10]We tried other settings, i.e. 25%, 50%, 100%, but in general we achieve best result when we use 75% of the training data.

Language	Input	Dev	Test
Czech	char	91.2	90.6
	char (multi-task)	91.6	91.0
	char + predicted case	**92.2**	**91.8**
	char + gold case	92.3	91.9
	oracle	92.5	92.0
German	char	87.5	84.5
	char (multi-task)	**87.9**	84.4
	char + predicted case	87.8	**86.4**
	char + gold case	90.2	86.9
	oracle	89.7	86.5
Russian	char	91.6	92.4
	char (multi-task)	92.2	92.6
	char + predicted case	**92.5**	**93.3**
	char + gold case	92.8	93.5
	oracle	92.6	93.3

Table 4.6: LAS results when case information is added. We use **bold** to highlight the best results for models without explicit access to gold annotations.

cial. Curiously, the MTL parser is worse than the the pipeline parser, but the MTL case tagger is better than the pipeline case tagger (Table 4.7). This indicates that the MTL model must learn to encode case in the model's representation, but must not learn to effectively use it for parsing. Finally, we observe that augmenting the charLSTM with either gold or predicted case improves the parsing performance for all languages, and indeed closes the performance gap with the full oracle, which has access to *all* morphological features. This is especially interesting, because it shows using carefully targeted linguistic analyses can improve accuracy as much as wholesale linguistic analysis.

Language	%case	Dev		Test	
		PL	MT	PL	MT
Czech	66.5	95.4	**96.7**	95.2	**96.6**
German	36.2	**92.6**	92.0	90.8	**91.4**
Russian	55.8	95.8	**96.5**	95.9	**96.5**

Table 4.7: Case accuracy for case-annotated tokens, for pipeline (**PL**) vs. multitask (**MT**) setup. **%case** shows percentage of training tokens annotated with case.

4.5 Understanding head selection

The previous experiments condition their analysis on the *dependent*, but dependency is a relationship between dependents and heads. We also want to understand the importance of morphological features to the *head*. Which morphological features of the head are important to the oracle?

4.5.1 Composing features in the oracle

To see which morphological features the oracle depends on when making predictions, we augmented our model with a **gated attention mechanism** following Kuncoro et al. (2017). Our new model attends to the morphological features of candidate head w_j when computing its association with dependent w_i (Eq. 4.3), and morpheme representations are then scaled by their *attention weights* to produce a final representation.

Let $f_{i1}, \cdots, f_{ik}$ be the k morphological features of w_i, and denote by $\mathbf{f}_{i1}, \cdots, \mathbf{f}_{ik}$ their corresponding *feature embeddings*. As in §2.4, $\mathbf{h}_i$ and $\mathbf{h}_j$ are the encodings of w_i and w_j, respectively. The morphological representation $\mathbf{m}_j$ of w_j is:

$$\mathbf{m}_j = [\mathbf{f}_{j1}, \cdots, \mathbf{f}_{jk}]^\top \mathbf{k} \tag{4.7}$$

where $\mathbf{k}$ is a vector of attention weights:

$$\mathbf{k} = \text{softmax}([\mathbf{f}_{j1}, \cdots, \mathbf{f}_{jk}]^\top \mathbf{V}\mathbf{h}_i) \tag{4.8}$$

The intuition is that dependent w_i can choose which morphological features of w_j are most important when deciding whether w_j is its head. Note that this model is asymmetric: a word only attends to the morphological features of its (single) parent, and not its (many) children, which may have different functions. [11]

We combine the morphological representation with the word's encoding via a sigmoid gating mechanism.

$$\mathbf{z}_j = \mathbf{g} \odot \mathbf{h}_j + (1 - \mathbf{g}) \odot \mathbf{m}_j \tag{4.9}$$

$$\mathbf{g} = \sigma(\mathbf{W}_1 \mathbf{h}_j + \mathbf{W}_2 \mathbf{m}_j) \tag{4.10}$$

where $\odot$ denotes element-wise multiplication. The gating mechanism allows the model to choose between the computed word representation and the weighted morphological representations, since for some dependencies, morphological features of the head might not be important. In the final model, we replace Equation 4.3 and Equation 4.4 with the following:

$$P_{head}(w_j | w_i, w) = \frac{\exp(a(\mathbf{h}_i, \mathbf{z}_j))}{\sum_{j'=0}^{N} \exp a(\mathbf{h}_i, \mathbf{z}_{j'})} \tag{4.11}$$

$$a(\mathbf{h}_i, \mathbf{z}_j) = \mathbf{v}_a \tanh(\mathbf{U}_a \mathbf{h}_i + \mathbf{W}_a \mathbf{z}_j) \tag{4.12}$$

The modified label prediction is:

$$P_{label}(\ell_k | w_i, w_j, w) = \frac{\exp(f(\mathbf{h}_i, \mathbf{z}_j)[k])}{\sum_{k'=0}^{|L|} \exp(f(\mathbf{h}_i, \mathbf{z}_j)[k'])} \tag{4.13}$$

where f is again a function to compute label score:

$$f(\mathbf{h}_i, \mathbf{z}_j) = \mathbf{V}_\ell \tanh(\mathbf{U}_\ell \mathbf{h}_i + \mathbf{W}_\ell \mathbf{z}_j) \tag{4.14}$$

4.5.2 Attention to headword morphological features

We trained our augmented model (**oracle-attn**) on Finnish, German, Czech, and Russian. Its accuracy is very similar to the oracle model (Table 4.8), so we obtain a more interpretable model with no change to our main results.

Next, we look at the learned attention vectors to understand which morphological features are important, focusing on the core arguments: nominal subjects, objects, and indirect objects. Since our model knows the case of each dependent, this enables us to

[11] This is a simple and much less computationally demanding variant of the model of Dozat et al. (2017), which uses different views for each head/dependent role.

Language	oracle		oracle-attn	
	UAS	LAS	UAS	LAS
Finnish	**89.2**	**87.3**	88.9	86.9
Czech	93.4	91.3	**93.5**	**91.3**
German	90.4	88.7	**90.7**	**89.1**
Russian	**93.9**	**92.8**	93.8	92.7

Table 4.8: Our attention experiment results on development set.

understand what features it seeks in potential heads for each case. For simplicity, we only report results for words where both head and label predictions are correct.

Figure 4.4 shows how attention is distributed across multiple features of the head word. In Czech and Russian, we observe that the model attends to *Gender* and *Number* when the noun is in nominative case. This makes intuitive sense since these features often signal subject-verb agreement. As we saw in earlier experiments, these are features for which a character model can learn reliably good representations. For most other dependencies (and all dependencies in German), *Lemma* is the most important feature, suggesting a strong reliance on lexical semantics of nouns and verbs. However, we also notice that the model sometimes attends to features like *Aspect*, *Polarity*, and *VerbForm*—since these features are present only on verbs, we suspect that the model may simply use them as convenient signals that a word is verb, and thus a likely head for a given noun.

4.6 Conclusions

In Chapter 1, we stated our main research questions on whether character-level models learn morphology and on whether we still need to explicitly model morphology. Character-level models are effective because they can represent OOV words and orthographic regularities of words that are consistent with morphology. But they depend on context to disambiguate words, and for some words this context is insufficient. *Case syncretism* is a specific example that our analysis identified, but the main results in Table 4.2 hint at the possibility that different phenomena are at play in different languages.

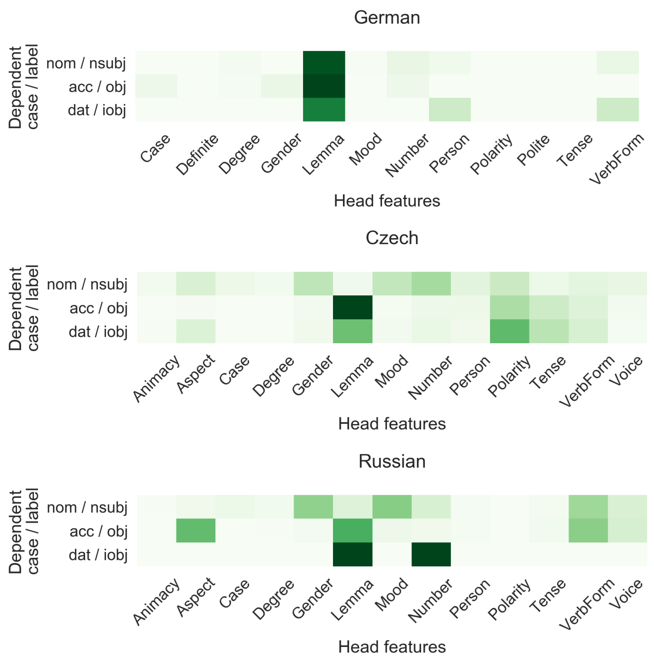


Figure 4.4: The importance of morphological features of the head for subject and object predictions.

While our results show that prior knowledge of morphology is important, they also show that it can be used in a targeted way: our character-level models improved markedly when we augmented them only with case. This suggests a pragmatic reality in the middle of the wide spectrum between pure machine learning from raw text input and linguistically-intensive modeling: our new models don't need all prior linguistic knowledge, but they clearly benefit from some knowledge in addition to raw input. While we used a data-driven analysis to identify case syncretism as a problem for neural parsers, this result is consistent with previous linguistically-informed analyses (Seeker and Kuhn, 2013; Tsarfaty et al., 2010). We conclude that neural models can still benefit from linguistic analyses that target specific phenomena where annotation is likely to be useful.

Chapter 5

Character-Level Models in Cross-Lingual, Low-Resource NLP

In Chapter 1, we listed a claim from Lee et al. (2017) that character-level models are useful for multilingual setting because they can easily identify morphemes that are shared across languages with overlapping alphabets. Indeed, several studies have shown that cross-lingual models that share character-level inputs tend to outperform monolingual models (Kann et al., 2017; Cotterell and Heigold, 2017; de Lhoneux et al., 2018), especially when they are trained on related languages. In Chapters 3 and 4, we have also seen the effectiveness of character-level models trained for monolingual models. The main focus of this chapter is to investigate whether there is a morphological effect when we train cross-lingual models that share character-level inputs. Cross-lingual models have proven to benefits low-resource NLP (Johnson et al., 2017; Cotterell and Heigold, 2017), and in this study we will investigate the morphological effects of character-level models for cross-lingual, low-resource dependency parsing. We experiment with parsing strategies which leverage both character-level inputs and linguistic annotations: (1) data augmentation, (2) cross-lingual training, and (3) transliteration. We show how our strategies, particularly cross-lingual training improves low-resource parsing. However, our analysis suggests that the improvements are mostly come from the structural similarities between languages and we do not yet find any strong evidence of morphological transfer in the cross-lingual training. This study was conducted in collaboration with Yova Kementchedjhieva and Anders Søgaard, which helped in the research discussions.

5.1 Motivation

Large annotated treebanks are available for only a tiny fraction of the worlds languages, and there is a wealth of literature on strategies for parsing with few resources (Hwa et al., 2005; Zeman and Resnik, 2008; McDonald et al., 2011; Søgaard, 2011). A popular approach is to train a parser on a related high-resource language and adapt it to the low-resource language. This approach benefits from the availability of Universal Dependencies (UD; Nivre et al., 2016), prompting substantial research (Tiedemann and Agic, 2016; Agić, 2017; Rosa and Mareček, 2018), along with the VarDial and the CoNLL UD shared tasks (Zampieri et al., 2017; Zeman et al., 2017b, 2018).

But low-resource parsing is still difficult. The organizers of the CoNLL 2018 UD shared task (Zeman et al., 2018) report that, in general, results on the task's nine low-resource treebanks "are extremely low and the outputs are hardly useful for downstream applications. So if we want to build a parser in a language with few resources, what can we do? To answer this question, we systematically compare several practical strategies for low-resource parsing, asking:

1. What can we do with only a very small *target* treebank for a low-resource language?
2. What can we do if we also have a *source* treebank for a related high-resource language?
3. What if the source and target treebanks do not share a writing system?

Each of these scenarios requires different approaches. **Data augmentation** is applicable in all scenarios, and has proven useful for low-resource NLP in general (Fadaee et al., 2017; Bergmanis et al., 2017; Sahin and Steedman, 2018b). Transfer learning via **cross-lingual training** is applicable in scenarios 2 and 3. Finally, **transliteration** may be useful in scenario 3.

To keep our scenarios as realistic as possible, we assume that no taggers are available since this would entail substantial annotation. Therefore, our neural parsing models must learn to parse from words or characters—that is, they must be **lexicalized**—even though there may be little shared vocabulary between source and target treebanks. While this may intuitively seem to make cross-lingual training difficult, recent results have shown that lexical parameter sharing on characters and words can in fact improve cross-lingual parsing (de Lhoneux et al., 2018) and that in some circumstances, a lexicalized parser can outperform a delexicalized one, even in a low-resource setting (Falenska and Çetinoğlu, 2017). This assumption also allows us to test the claim from Lee et al. (2017) that character-level models can identify morphemes that are shared across different languages with overlapping alphabets.

We experiment on three language pairs from different language families, in which the first of each is a genuinely low-resource language: North Sámi and Finnish (Uralic); Galician and Portuguese (Romance); and Kazakh and Turkish (Turkic), which have different writing systems. To avoid optimistic evaluation, we extensively experiment only with North Sámi, which we also analyse to understand *why* our cross-lingual training outperforms the other parsing strategies. We treat Galician and Kazakh as truly held-out, and test only our best methods on these languages. Our results show that:

1. When no source treebank is available, data augmentation is very helpful: dependency tree morphing improves labeled attachment score (LAS) by as much as 9.3%. Our analysis suggests that syntactic rather than lexical variation is most useful for data augmentation.

2. When a source treebank is available, cross-lingual parsing improves LAS up to 16.2%, but data augmentation still helps, by an additional 2.6%. Our analysis suggests that improvements from cross-lingual parsing occur because the parser learns syntactic regularities about word order, since it does not have access to POS and has little reusable information about word forms.

3. If source and target treebanks have different writing systems, transliterating them to a common orthography is very effective.

5.2 Methods

We describe three techniques for improving low-resource parsing: (1) two data augmentation methods which have not been applied before for dependency parsing, (2) cross-lingual training, and (3) transliteration.

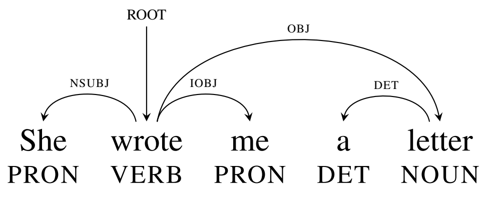
5.2.1 Data augmentation by dependency tree morphing (TreeMorph)

Sahin and Steedman (2018b) introduce two label-preserving operations to augment a dataset for low-resource POS tagging. Their method assumes access to a dependency tree, but they do not test it for dependency parsing, which we do here for the first time. The first operation, *cropping*, removes some parts of a sentence to extract a smaller or simpler, meaningful sentence. The second operation, *rotation*, keeps all the words in the sentence but re-orders flexible tree fragments attached to the *root* verb. Here, the flexible fragments are defined as subtrees, which are flexible to move around (verb, subject, or object). Since flexibility depends on the morphological typology of the language, rotation would make more sense for language with extensive marking system which have no strict word order. For languages close to analytical typology such as English or Mandarin, rotation would mostly introduce noise. Figure 5.1 demonstrate how cropping and rotation are applied to a sentence *'She wrote me a letter'*. Figure 5.1b crops the direct object, to focus on the subject and object. Figure 5.1c rotates the indirect object ‘me’ and put it after the subject ‘she’.

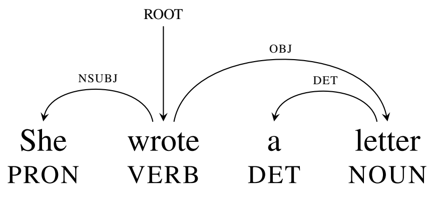
To extract the flexible fragments, Sahin and Steedman (2018b) define the Label of Interest (LOI) which consists of NSUBJ (nominal subject), OBJ (direct object), IOBJ (indirect object), or OBL (oblique nominal) dependencies. To address cases when the root verb is a phrase, the following relations are used: FIXED, FLAT, COP (copula), and COMPOUND.

It is important to note that although both operations change the set of words or the word order, they do not change the dependencies. Hence, they provide the model with more input examples. While these may be awkward or ill-formed, the corresponding analyses are still likely to be correct, and thus beneficial for learning.

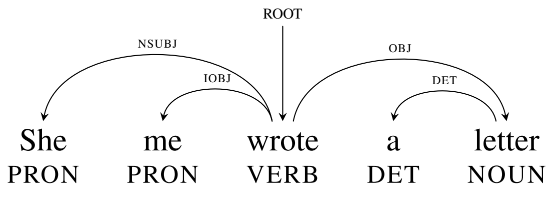
Cropping and rotation might change the sentence-level meaning, but they also allow the model to learn variations in argument structure (cropping) and variability in constituent order (rotation), which may benefit languages with flexible word order and rich morphology. Some of our low-resource languages have these properties—while North Sámi has a fixed word order (SVO), Galician and Kazakh have relatively free word order. All the three languages use a case marking on nouns, which means that word order may not be important for correct attachment.



(a) Original sentence.



(b) Cropped sentence.



(c) Rotated sentence.

Figure 5.1: Examples of dependency tree morphing operations.

Both rotation and cropping can produce many trees. We use the default parameters given in Sahin and Steedman (2018b).

5.2.2 Data augmentation by nonce sentence generation (Nonce)

Our next data augmentation method is adapted from Gulordava et al. (2018). The main idea is to create *nonce* sentences by replacing some of the words which have the same syntactic annotations. For each training sentence, we replace each content word—nouns, verbs, or adjective—with an alternative word having the same universal POS and morphological features. To preserve dependencies, we also introduce an additional constraint that the replacement word should also have the same dependency label.[1] Specifically, for each content word, we first stochastically choose whether to replace it; then, if we have chosen to replace it, we uniformly sample the replacement word type meeting the corresponding constraints. For instance, given a sentence "*He*

[1]The dependency label constraint is not used by Gulordava et al. (2018).

borrowed a book from the library.", we can generate the following sentences:

(11) a. He bought a book from the shop .

b. He wore a umbrella from the library .

This generation method is only based on syntactic features (i.e., morphology and dependency labels), so it sometimes produces nonsensical or ungrammatical sentences like 11b. But since we only replace words if they have the same morphological features and dependency label, this method preserves the original tree structures in the treebank. Following (Gulordava et al., 2018), we generate five nonce sentences for each original sentence.

5.2.3 Cross-lingual training

When a source treebank is available, model transfer is a viable option. We perform model transfer by cross-lingual parser training: we first train on both source and target treebanks to produce a single model, and then fine tune the model only on the target treebank. In our preliminary experiments, the fine-tuning strategy gives us the best results on development sets (Appendix A.2).

5.2.4 Transliteration

Two related languages might not share a writing system even when they belong to the same family. We evaluate whether a simple transliteration would be helpful for cross-lingual training in this case. In our study, the Turkish treebank is encoded in extended Latin script while the Kazakh treebank is encoded in Cyrillic script. This difference makes model transfer less useful, and means we might not be able to leverage lexical similarities between the two languages. We pre-process both treebanks by transliterating them to the same “pivot” alphabet, basic Latin.[2]

The mapping from Turkish is straightforward. Its alphabet consists of 29 letters, 23 of which are in basic Latin. The other six letters, i.e., ‘ç’,‘’, ‘’, ‘ö’, ‘’, ‘ü’ are modifications of their non-diacritic Latin counterparts, facilitating different pronunciations.[3] We map these characters to the basic Latin characters, e.g., ‘ç’ to ‘c’. For Kazakh, we use

[2]Another possible pivot is phonemes (Tsvetkov et al., 2016). We leave this as future work.

[3]https://www.omniglot.com/writing/turkish.htm

a simple dictionary created by a Kazakh computational linguist to map each Cyrillic letter to the basic Latin alphabet. The mapping from Kazakh Cyrilic into basic Latin alphabet is provided in Appendix A.3.

5.3 Experimental Setup

5.3.1 Dependency Parsing Model

We use the Uppsala parser, a transition-based neural dependency parser (de Lhoneux et al., 2017a,b; Kiperwasser and Goldberg, 2016). The parser uses an arc-hybrid transition system (Kuhlmann et al., 2011), extended with a static-dynamic oracle and SWAP transition to allow non-projective dependency trees (Nivre, 2009).

Let $w = w_0, \ldots, w_{|w|}$ be an input sentence of length $|w|$ and let w_0 represent an artificial ROOT token. We create a vector representation for each input token w_i by concatenating $(;)$ its word embedding, $\mathbf{e}_w(w_i)$ and its character-based word embedding, $\mathbf{e}_c(w_i)$:

$$\mathbf{x}_i = [\mathbf{e}_w(w_i); \mathbf{e}_c(w_i)] \tag{5.1}$$

Here, $\mathbf{e}_c(w_i)$ is the output of a *character-level* bidirectional LSTM (biLSTM) encoder run over the characters of w_i Ling et al. (2015a); this makes the model fully open-vocabulary, since it can produce representations for any character sequence. We then obtain a *context-sensitive* encoding $\mathbf{h}_i$ using a *word-level* biLSTM encoder:

$$\mathbf{h}_i = [\text{LSTM}_f(\mathbf{x}_{0:i}); \text{LSTM}_b(\mathbf{x}_{|w|:i})] \tag{5.2}$$

We then create a configuration by concatenating the encoding of a fixed number of words on the top of the stack and the beginning of the buffer. Given this configuration, we predict a transition and its arc label using a multi layer perceptron (MLP). More details of the core parser can be found in de Lhoneux et al. (2017a,b).

5.3.2 Parameter sharing

To train cross-lingual models, we use the strategy of de Lhoneux et al. (2018) for parameter sharing, which uses *soft* sharing for word and character parameters, and *hard* sharing for the MLP parameters. Soft parameter sharing uses a language embedding, which, in theory, learns what parameters to share between the two languages. Let $\mathbf{c}_j$ be an embedding of character c_j in a token w_i from the treebank of language k, and let $\mathbf{l}_k$ be the language embedding. For sharing on characters, we concatenate character and language embedding: $[\mathbf{c}_j; \mathbf{l}_k]$ for input to the character-level biLSTM. Similarly, for input to the word-level biLSTM, we concatenate the language embedding to the word embedding, modifying Eq. 5.1 to

$$\mathbf{x}_i = [\mathbf{e}_w(w_i); \mathbf{e}_c(w_i); \mathbf{l}_k] \tag{5.3}$$

We use the default hyperparameters of de Lhoneux et al. (2018) in our experiments. We fine-tune each model by training it further only on the target treebank (Shi et al., 2016). We use early stopping based on Label Attachment Score (LAS) on development set.

5.3.3 Datasets

We use Universal Dependencies (UD) treebanks version 2.2 Nivre et al. (2018). Our target treebanks are North Sámi Giella (Sheyanova and Tyers, 2017), Galician TreeGal, and Kazakh KTB (Tyers and Washington, 2015; Makazhanov et al., 2015). None of these treebanks have a development set, so we generate new train/dev splits by 50:50 (Table 5.1). Having large development sets allow us to perform better analysis for this study. We use Finish TDT, Portuguese Bosque (Rademaker et al., 2017), and Turkish IMST for our source treebanks. Similar to our preprocessing steps in Chapter 4, we ignore multiword tokens and empty nodes (for ellipsis) in the treebanks, and use gold sentence segmentation and tokenization provided in UD. Table 5.1 shows the statistics of our datasets.

5.4 Parsing North Sámi

North Sámi is our largest low-resource treebank, so we use it for a full evaluation and analysis of different strategies before testing on the other languages. To understand the effect of target treebank size, we generate three datasets with different *training* sizes: $\mathcal{T}_{10}$ (~10%), $\mathcal{T}_{50}$ (~50%), and $\mathcal{T}_{100}$ (100%). Table 5.2 reports the number of training sentences after we augment the data using methods described in Section 5.2.

Language	train	dev.	test
Finnish	14981	1875	1555
North Sámi	1128	1129	865
Portuguese	8329	560	477
Galician	300	300	400
Turkish	3685	975	975
Kazakh	15	16	1047

Table 5.1: Train/dev split used for each treebank.

	original	+TreeMorph	+Nonce
$\mathcal{T}_{100}$	1128	7636	4934
$\mathcal{T}_{50}$	564	3838	2700
$\mathcal{T}_{10}$	141	854	661

Table 5.2: Number of North Sámi training sentences.

We employ the following baselines: unsupervised left-branching and right-branching, word-level only model, monolingual, and cross-lingual models, all without data augmentation. We use unsupervised and word-level only baselines to test whether character-level features are still useful in low-resource settings. The monolingual model acts as a simple baseline, to resemble a situation when the target treebank does not have any source treebank (i.e., no available treebanks from related languages). The cross-lingual model serves as a strong baseline, simulating a case when there is a source treebank.

Table 5.3 shows our results. Unsupervised models obtain very poor performance, with 17.5% and 24.2% UAS for right-branching and left-branching, respectively. Models with word-level inputs outperform the unsupervised models, but consistently worse than models with word and character-level inputs. This result suggests that character-level features are still useful in addition to word-level information, showing their potential to handle rare and OOV words, which is also related to morphology. Next, we discuss our results based on our motivating scenarios of low-resource dependency parsing.

		$\mathcal{T}_{10}$		$\mathcal{T}_{50}$		$\mathcal{T}_{100}$	
Input	Model	UAS	LAS	UAS	LAS	UAS	LAS
unsupervised	right-branching	17.5	-	17.5	-	17.5	-
	left-branching	24.2	-	24.2	-	24.2	-
word	monolingual	28.4	14.5	48.9	35.9	57.7	46.5
word+char	monolingual	35.5	18.5	54.5	42.5	63.9	53.3
	monolingual+TreeMorph	41.0	27.1	56.9	46.6	65.4	56.0
	monolingual+Nonce	42.9	27.8	56.9	46.5	65.4	56.3
word	multilingual	41.9	28.3	54.3	42.7	61.3	51.2
word+char	cross-lingual	47.2	34.7	**61.0**	51.7	**69.1**	61.3
	cross-lingual+TreeMorph	**48.9**	**37.3**	60.2	**52.0**	68.5	60.9
	cross-lingual+Nonce	47.5	35.4	60.1	**52.0**	**69.1**	**61.7**

Table 5.3: Parsing results on North Sámi on development data.

Scenario 1: we only have a very small target treebank. In the monolingual experiments, we observe that both dependency tree morphing (TREEMORPH) and nonce sentence generation (NONCE) improve performance, indicating the strong benefits of data augmentation when there is no other resources available except the target treebank itself. In particular, when the number of training data is the lowest ($\mathcal{T}_{10}$), data augmentations improves performance up to 9.3% LAS.

Scenario 2: a source treebank is available. We see that the cross-lingual training (cross-base) performs better than monolingual models even with augmentation. For the $\mathcal{T}_{10}$ setting, cross-base achieves almost twice as much as the monolingual baseline (mono-base). The benefits of data augmentation are less evident in the cross-lingual setting, but in the $\mathcal{T}_{10}$ scenario, data augmentation still clearly helps. Overall, cross-lingual combined with TREEMORPH yields the best result.

5.4.1 What is learned from Finnish?

Why do cross-lingual training and data augmentation help? To put this question in context, we first consider their relationship. Finnish and North Sámi are mutually unintelligible, but they are typologically similar: of the 49 (mostly syntactic) linguistic features annotated for North Sámi in the Word Atlas of Languages (WALS; Dryer and Haspelmath, 2013), Finnish shares the same values for 42 of them.[4] Despite this and their phylogenetic and geographical relatedness, they share very little vocabulary: only 6.5% of North Sámi tokens appear in Finnish data, and these words are either proper nouns or closed class words such as pronouns or conjunctions. However, both languages do share many character-trigrams (72.5%, token-level), especially on suffixes.

Now we turn to an analysis of the $\mathcal{T}_{10}$ data setting, where we see the largest gains for all methods.

5.4.2 Analysis of data augmentation

For dependency parsing, POS features are important because they can provide strong signals whether there exists dependency between two words in a given sentence. For example, *subject* and *object* dependencies often occur between a NOUN and a VERB, as can be seen in Fig. 5.1a. We investigate the extent to which data augmentation is useful for learning POS features, using diagnostic classifiers (Veldhoen et al., 2016b; Adi et al., 2017; Shi et al., 2016) to probe our model representations. Our central question is: do the models learn useful representations of POS, despite having no direct access to it? And if so, is this helped by data augmentation?

After training each model, we freeze the parameters and generate *context-dependent* representations (i.e., the output of *word-level* biLSTM, $\mathbf{h}_i$ in Eq. 5.2), for the training and development data. We then train a feed-forward neural network classifier to predict the POS tag of each word, using only the representation as input. To filter out the effect of cross-lingual training, we only analyze representations trained using the *monolingual* models. Our training and development data consists of 6321 and 7710 tokens, respectively. The percentage of OOV tokens is 40.5%.

[4]There are 192 linguistic features in WALS, but only 49 are defined for North Sámi. These features are mostly syntactic, annotated within different areas such as morphology, phonology, nominal and verbal categories, and word order.

POS	%dev	baseline	%diff. with	
			+TreeMorph	+Nonce
INTJ	0.1	0.0	20.0	20.0
PART	1.5	70.1	7.7	0.8
NUM	1.9	19.2	15.1	-4.1
ADP	1.9	15.7	24.5	19.7
SCONJ	2.4	57.8	5.9	7.6
AUX	3.2	26.3	27.2	-4.9
CCONJ	3.4	91.3	-0.8	-4.2
PROPN	4.7	5.9	5.9	-5.9
ADJ	6.5	12.7	3.8	0.2
ADV	9.0	42.9	11.8	11.5
PRON	13.4	63.2	5.4	-2.7
VERB	25.7	72.4	-6.2	-4.5
NOUN	26.4	67.0	8.6	13.2

Table 5.4: Results for the monolingual POS predictions, ordered by the frequency of each tag in the dev split (%dev). %diff shows the difference between each augmentation method and monolingual models.

Table 5.4 reports the POS prediction accuracy. We observe that representations generated with monolingual TreeMorph seem to learn better POS, for most of the tags. On the other hand, representations generated with monolingual Nonce sometimes produce lower accuracy on some tags; only on nouns the accuracy is better than monolingual TreeMorph. We hypothesize that this is because Nonce sometimes generates meaningless sentences which confuse the model. In parsing this effect is less apparent, mainly because monolingual Nonce has the poorest POS representation for infrequent tags (%dev), and better representation of nouns.

5.4.3 Effects of cross-lingual training

Next, we analyze the effect of cross-lingual training by comparing the monolingual baseline to the cross-lingual model with TreeMorph.

	Top nearest Finnish words	
North Sámi	char-level	word-level
borrat (VERB; eat)	*herrat* (NOUN; gentleman)	*käydä* (VERB; go)
	kerrat (NOUN; time)	*otan* (VERB; take)
	naurat (VERB; laugh)	*sain* (VERB; get)
veahki (NOUN; help)	*nuuhki* (VERB; sniff)	*tyhjäksi* (ADJ; empty)
	väki (NOUN; power)	*johonki* (PRON; something)
	avarsi (VERB; expand)	*lähtökohdaksi* (NOUN; basis)
divrras (ADJ; expensive)	*harras* (ADJ; devout)	*välttämätöntä* (ADJ; essential)
	reipas (ADJ; brave)	*mahdollista* (ADJ; possible)
	sarjaporras (NOUN; series)	*kilpailukykyisempi* (ADJ; competitive)

Table 5.5: Most similar Finnish words for each North Sámi word based on cosine similarity.

Cross-lingual representations. The fact that cross-lingual model improves parsing performance is interesting, since Finnish and North Sámi have so little common vocabulary. What linguistic knowledge is transferred through cross-lingual training? We analyze whether words with the same POS category from the source and target treebanks have similar representations. To do this, we analyze the *head predictions*, and collect North Sámi tokens for which only the cross-lingual model correctly predicts the headword.[5] For these words, we compare token-level representations of North Sámi *development* data to Finnish *training* data.

We ask the following questions: Given the representation of a North Sámi word, what is the Finnish word with the most similar representation? Do they share the same POS category? Information other than POS may very well be captured, but we expect that the representations will reflect similar POS since POS is highly revelant to parsing. We use *cosine distance* to measure similarity.

We look at four categories for which cross-lingual training substantially improves results on the development set: adjectives, nouns, pronouns, and verbs. We analyze representations generated by two layers of the model in §5.3.1: (1) the output of

[5]Another possible way is to look at the label predictions. But since the monolingual baseline LAS is very low, we focus on the unlabeled attachment prediction since it is more accurate.

POS	char-level (%)	word-level (%)
ADJ	12.1	37.1
NOUN	55.8	63.5
PRON	12.9	68.0
VERB	34.2	69.0

Table 5.6: # of North Sámi tokens for which the most similar Finnish word has the same POS.

character-level biLSTM (char-level), $\mathbf{e}_c(w_i)$ and (2) the output of word-level biLSTM (word-level), i.e., $\mathbf{h}_i$ in Eq. 5.2.

Table 5.5 shows examples of top three closest Finnish training words for a given North Sámi word. We observe that character-level representation focuses on orthographic similarity of suffixes, rather than POS. On the word-level representations, we find more cases when the top closest Finnish words have the same POS with the North Sámi word. In fact, when we compare the most similar Finnish word (Table 5.6) quantitatively, we find that the word-level representations of North Sámi are often similar to Finnish word with the same POS; the same trend does not hold for character-level representations. Since very few word tokens are shared, this suggests that improvements in cross-lingual training might simply be due to syntactic (i.e. word order) similarities between the two languages, captured in the dynamics of the biLSTM encoder—despite the fact that it knows very little about the North Sámi tokens themselves. The word-level representation has advantage over the char-level representation in the way that it has access to contextual information like word order, and it has knowledge about the other words in the sentence.

Head and label prediction. Lastly, we analyze the parsing performance of the monolingual compared to the cross-lingual models. Looking at the produced parse trees, one striking difference is that monolingual model sometimes predicts a “rootless" tree, i.e., it fails to label any word in the sentence with a *root* label. In cases where the monolingual model predicts wrong parses and the cross-lingual model predicts the correct ones, we find that the “rootless" trees are predicted more than 50% of the time.[6] Mean-

[6]The parsing model enforces the constraint that every tree should have a head, i.e., an arc pointing from a dummy root to a node in the tree. It does not, however, enforce that this arc be labeled *root*—the

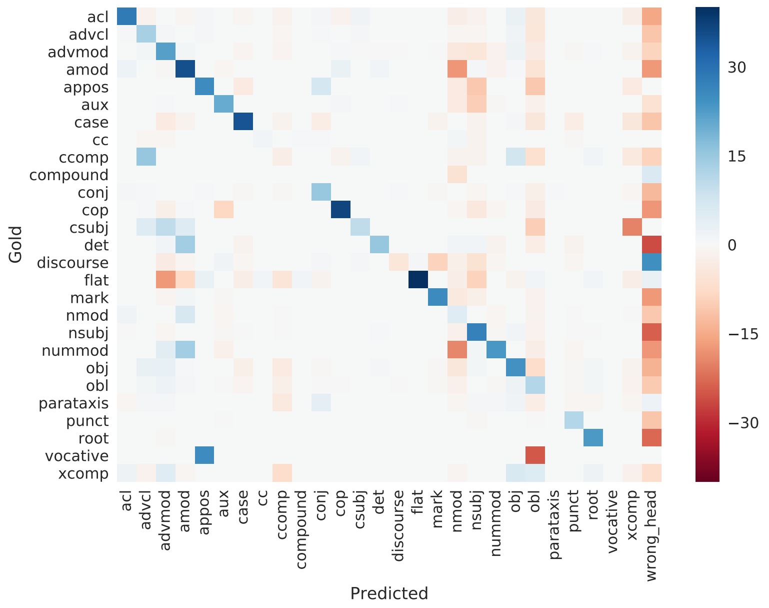


Figure 5.2: Differences between cross-lingual vs. monolingual confusion matrices. The last column represents cases of *incorrect* heads and the other columns represent cases for *correct* heads, i.e., each row summing to 100%. Blue cells show higher cross-lingual values and red cells show higher monolingual values.

while, the cross-lingual model always predicts a root for every sentence, even when the predicted head index is incorrect (i.e., root must always be at index 0). This pattern suggests that more training examples at least helps the model to learn structural properties of a well-formed tree.

The ability of a parser to predict labels is contingent on its ability to predict heads, so we focus our analysis on two cases. How do monolingual and cross-lingual head prediction compare? And if both models predict the correct head, how do they compare on label prediction?

Figure 5.2 shows the *difference* between two confusion matrices: one for cross-lingual and one for monolingual models. The last column shows cases of *incorrect* heads and the other columns show label predictions when the heads are *correct*, i.e., each row summing to 100%. Here, blue cells highlight confusions that are more common for the cross-lingual model, while red cells highlight those more common for the monolingual model. For head prediction (last column), we observe that monolingual model makes higher errors especially for nominals and modifier words. In cases when both both

model must learn the labeling.

		CROSS-LINGUAL	
Language	zero-shot	+fastText	+TreeMorph
Galician	51.9	**72.8**	71.0
Kazakh	12.5	27.7	**28.4**
Kazakh (translit.)	21.2	31.1	**36.7**

Table 5.7: LAS results on **development sets**. *zero-shot* denotes results where we predict using model trained only on the source treebank.

			CROSS-LINGUAL		
	baseline	best system	+fastText	+TreeMorph	rank
Galician	66.16	74.25	**70.46**	69.21	10/27
Kazakh (translit.)	24.21	31.93	25.28	**28.23**	2/27

Table 5.8: Comparison to CoNLL 2018 UD Shared Task on **test sets**. *best system* is the state-of-the-art model for each treebank: UDPipe-Future (Straka, 2018) for Galician and Uppsala (Smith et al., 2018a) for Kazakh. *rank* shows our best model position in the shared task ranking for each treebank.

models predict the correct heads, we observe that cross-lingual training gives further improvements in predicting most of the labels. In particular, regarding the “rootless" trees discussed before, we see evidence that cross-lingual training helps in predicting the correct root index, and the correct *root* label.

5.5 Parsing truly low-resource languages

Now we turn to two truly low-resource treebanks: Galician and Kazakh. These treebanks are most analogous to the North Sámi $\mathcal{T}_{10}$ setting and therefore we apply the best approach, cross-lingual training with TREEMORPH augmentation. Table 5.1 provides the statistics of the augmented data. For Galician, we use the Portuguese treebank as source while for Kazakh we use Turkish. Portuguese and Galician have high vocabulary overlap; 62.9% of Galician tokens appear in Portuguese data, and they share while for Turkish and Kazakh they do not share vocabulary since they use different writing systems. However, after transliterating them into the same basic Latin alphabet, we observe that 9.5% of Kazakh tokens appear in the Turkish data. Both language pairs also share many (token-level) character trigrams: 96.0% for Galician-Portuguese and 66.3% for transliterated Kazakh-Turkish.

To compare our best approach, we create two baselines: (1) a pre-trained parsing model of the source treebank (zero-shot learning), and (2) a cross-lingual model initialized with *monolingual* pre-trained word embeddings. The first serves as a weak baseline, in a case where training on the target treebank is not possible (e.g., Kazakh only has 15 sentences for training). The latter serves as a strong baseline, in a case when we have access to pre-trained word embeddings, for the source and/or the target languages.

We treat a pre-trained word embedding as an external embedding, and concatenate it with the other representations, i.e., modifying Eq. 5.3 to:

$$\mathbf{x}_i = [\mathbf{e}_w(w_i); \mathbf{e}_p(w_i); \mathbf{e}_c(w_i); \mathbf{l}_k] \tag{5.4}$$

where $\mathbf{e}_p(w_i)$ represents a pre-trained word embedding of w_i, which we update during training. We use the pre-trained monolingual fastText embeddings Bojanowski et al. (2017).[7] We concatenate the source and target pre-trained word embeddings.[8] For our experiments with transliteration (§5.2.4), we transliterate the entries of both the source and the target pre-trained word embeddings.

5.5.1 Experiment results

Table 5.7 reports the LAS performance on the development sets. TreeMorph augmentation improves performance over the zero-shot baseline and achieves comparable or better LAS with a cross-lingual model trained with pre-trained word embeddings.

Next, we look at the effects of transliteration (see Kazakh vs Kazakh (translit.) in Table 5.7). In the zero-shot experiments, simply mapping both Turkish and Kazakh characters to the Latin alphabet improves accuracy from 12.5 to 21.2 LAS. Cross-lingual training with TreeMorph further improves performance to 36.7 LAS.

[7]The embeddings are available at https://fasttext.cc/docs/en/pretrained-vectors.html.

[8]If a word occurs in both source and target, we use the word embedding of the source language.

5.5.2 Comparison with CoNLL 2018

To see how our best approach (i.e., cross-lingual model with TREEMORPH augmentation) compares with the current state-of-the-art models, we compare it to the recent results from CoNLL 2018 shared task. Training state-the-art models may require lots of engineering and data resources. Our goal, however, is not to achieve the best performance, but rather to systematically investigate how far simple approaches can take us. We report performance of the following: (1) the shared task baseline model (UDPipe v1.2; Straka and Straková, 2017) and (2) the best system for each treebank, (3) our best approach, and (4) a cross-lingual model with fastText embeddings.

Table 5.8 presents the overall comparison on the test sets. For each treebank, we apply the same sentence segmentation and tokenization used by each best system.[9] We see that our approach outperforms the baseline models on both languages. For Kazakh, our model (with transliteration) achieves a competitive LAS (28.23), which would be the second position in the shared task ranking.

5.6 Conclusions

In this chapter, we have presented various strategies for low-resource dependency parsing. We focus on extremely low-resource scenarios where external data or taggers are hardly available.

Our strategies make use of two kind of information which are available on any treebanks: lexical features such as words or characters and linguistic annotations. We demonstrate that in the extremely low-resource setting, data augmentation improves parsing performance both in monolingual and cross-lingual settings. We also show that transfer learning is possible with *lexicalized* parsers, using words and characters information. In addition, we discover that transfer learning between two languages with different writing systems is possible, and future work should consider transliteration for other language pairs.

While we have not exhausted all the possible techniques (e.g., use of external resources

[9]UD shared task only provides unsegmented (i.e., sentence-level and token-level) raw test data. However, participants were allowed to use predicted segmentation and tokenization provided by the baseline UDPipe model.

(Rasooli and Collins, 2017; Rosa and Mareček, 2018), predicted POS (Ammar et al., 2016), multiple source treebanks (Lim et al., 2018; Stymne et al., 2018)), we show that simple methods which leverage the linguistic annotations in the treebank can improve low-resource parsing. A possible future work would be to explore different augmentation methods, such as the use of *synthetic* source treebanks (Wang and Eisner, 2018) or contextualized language model (Peters et al., 2018; Howard and Ruder, 2018; Devlin et al., 2019) for scoring the augmented data (e.g., using perplexity).

Chapters 3 and 4 of this thesis were concerned about character-level models and their relations to morphology, in comparison to the word lookup model and models with access to gold morphological annotations. We conclude that although these models are more effective than the traditional word lookup model, they can still benefit from linguistic analyses. Both studies focused on settings when we have arguably large amount of data, where we can conveniently train monolingual model for each language. In this chapter, we looked at a different setting, where there is only limited amount of data. We tested how far character-level models can improve parsing both in monolingual and cross-lingual settings. Our results show that cross-lingual training are very effective for improving low-resource parsing and this is because of the syntactic similarities between related languages that are captured by the neural networks. We have not yet observed any evidence on morphological transfer, but our qualitative analysis hints that this is possible when the two languages use similar affixes to mark morphology.

Understanding what kind of morphological information are transferred across languages is non-trivial. Unlike POS or dependency labels, morphology typically requires more fine-grained annotations which can be different even between two closely related languages. For example, according to UD annotations, Finnish has 15 possible values for case, while North Sámi has 8 possible values. Sometimes, there can also be semantic differences which tied to the morphological feature. For example, ‘car’ in Spanish (*el coche*) is marked as masculine, while in French (*la voiture*) it is marked as feminine. Recently, there has been an effort in building a universal schema for morphology, called Universal Morphological Feature Schema (Unimorph; Sylak-Glassman, 2016) to act as an interlingua for inflectional morphology. Exploiting this resource would be another interesting future work to investigate morphological transfer across languages.

Chapter 6

Conclusions

People continually use their morphological knowledge to understand, memorize, and apply new words in their language. In Chapter 1 we mentioned how we expect NLP systems to have the same capabilities so that they can process words and understand human languages. Recent progress on computational NLP models using neural networks provides opportunities to build such systems. Instead of modeling words as atomic units, we can now model each word as a function of its subword units, such as characters. Humans process words and learn the morphology without explicit guidance, so we ask, can neural models also do the same thing? In other words, does NLP still benefit from prior knowledge of morphology or can they be replaced entirely by models of subword units? This thesis aims to understand whether subword unit models, specifically character-level models, learn morphology.

In Chapter 2, we discussed how languages vary in terms of their morphological processes, and one of our main goals is to understand how current subword unit models interact with the morphological typology. To do so, in this thesis, we performed our study on a total of 18 typologically diverse languages.

In Chapter 3, we started by presenting a systematic comparison of subword unit models, varying (1) the choice of subword unit, (2) the compositional function, and (3) their interaction with language typology. We established character-level models (either composed by biLSTM or CNN) as strong baselines for learning morphology. To understand whether our strong baselines capture morphology, we compared character-level models to an oracle with access to morphological analysis. We showed that character-level models do not yet match the oracle performance, indicating that prior knowledge of morphology is still important for neural models. This comparison between subword unit models and oracle has not been explored in the past, and we argue that this is important in order to answer whether prior knowledge about morphology is still important for our neural models.

If character-level models performance do not yet match the oracle performance, what do they actually learn about morphology? We explored this question in Chapter 4. We diagnosed both character-level and oracle models performance on dependency parsing; a task which benefits substantially from morphological knowledge. We showed *case syncretism* as a specific phenomena that is not well-captured by the character-level models, but our results hint at the possibility that different linguistic phenomena are at play in different languages. We then demonstrated how a targeted, explicit modeling of morphology can be useful in such cases. This suggests that our neural models do not need all prior linguistic knowledge, but they clearly benefit from some knowledge in addition to raw text input.

The claim of Lee et al. (2017) that we listed in Chapter 1 states that character-level models are useful in a multilingual setting because they can easily identify morphemes that are shared across languages with overlapping alphabets. We tested this claim in Chapter 5. We applied character-level models for low-resource dependency parsing, an applicable situation where multilingual models would be useful. We explored strategies which exploit both character-level input and linguistic annotations: (1) data augmentation, (2) cross-lingual training, and (3) transliteration. Our experiments on North Sami, Galician, and Kazakh demonstrated that all of the three strategies improve low-resource parsing. Our analysis on cross-lingual representation suggests that cross-lingual training helps because related languages share similar syntactic structure (word order) but we did not yet find any clear evidence of morphological transfer across languages. We also showed how transliteration into a shared orthographic spaces is very helpful when two related languages use different writing systems.

6.1 Future Work

There are many directions for future work. In this thesis, we focused on learning morphology using subword level input. In practice, we would also like to be able to generate well-formed, grammatical text since this is crucial for applications like machine translation, question answering, or summarization. Character-level models are effective for representing input words, but are they also effective for generating text? Many existing models apply subword level input but only a few has started to explore them for text generation (Ling et al., 2015b; Lee et al., 2017; Matthews et al., 2018). Future work might explore the effects of morphological typology and different data settings (size or domains) for generating text at the subword level.

This thesis uses internal representations to analyze neural models, but there are many other interesting avenues to understand the morphological learning of neural models. For example, we can track the RNN/LSTM cell dynamics of the neural models to understand how neural models acquire or learn a specific linguistic property such as POS or morphological feature (Kementchedjhieva and Lopez, 2018), or a linguistic phenomenon like subject-verb agreement (Linzen et al., 2016; Lakretz et al., 2019). Since languages vary in their morphological processes, it would also be useful to understand which features or phenomena are difficult to learn by the neural models and if inclusion of linguistic information can help the learning process. Understanding what neural networks learn about language (including morphology) has also become of great interest in the NLP community (Linzen et al., 2018, 2019).

We argue that prior knowledge of morphology is still important for neural models. In Chapter 4 we demonstrated this using two simple approaches of multi-task learning and predicted morphological analyses as additional input features. However, it is desirable to have a model that can exploit annotations when they are available and predict them when they are not available (Zhou and Neubig, 2017). We can also explore some variants of multi-task learning which (1) use the predicted features distributions of the auxiliary task and use it for the target task (Anastasopoulos and Chiang, 2018) or (2) learn strategies to first learn the important morphological features before gradually focus on the target NLP task (Kiperwasser and Ballesteros, 2018).

In Chapter 4, we showed how contextual information might be useful for learning morphological features. Most recently, several models have been proposed for learning contextual word representations (Peters et al., 2018; Devlin et al., 2019). These models apply multiple layers in their network to learn some linguistic abstraction starting from the low-level features such as POS to more abstract features like semantics or topics. Do these models learn better morphology than our simple language model? There are still many open questions regarding what these models learn about languages and how they interact with typologically diverse languages. Answering these questions would also be an interesting direction for future research.

Neural network based models have gained huge interests from the NLP community in the past few years due to their impressive performance on various benchmarks. We position our work on the side which argues that it is also important to be able to analyze what the representations that neural NLP models actually learn about language, and to gain insights on how to improve them. In this thesis, we specifically looked at morphology, but there are way many more aspects of languages that remain to be investigated if we want to achieve our ultimate goal in building systems that can understand human languages. We hope this thesis can provide insights and research directions for related future work.

Appendix A

Additional Materials

A.1 Universal Inflectional Features

Our oracle models in Chapter 4 use universal morphological features defined in the UD annotation guidelines. In our experiments, we only use the inflectional features (Table A.1), and in practice the values of each feature vary across languages. More details explanation can be found in Universal Dependencies guidelines.[1]

Category	Inflectional Feature
Nominal	Gender, Animacy, NounClass, Number, Case, Definite, Degree
Verbal	VerbForm, Mood, Tense, Aspect, Voice, Evident, Polarity
	Person, Polite, Clusivity

Table A.1: Universal inflectional features defined in UD version 2.0. The labels nominal and verbal are used as approximate categories only.

[1]http://universaldependencies.org/u/feat/index.html

A.2 Effects of Fine-Tuning for Cross-Lingual Training

For our cross-lingual experiments in Chapter 5, we observe that fine-tuning on the target treebank always improves parsing performance. Table A.2 reports LAS for cross-lingual models with and without fine-tuning.

size	mono-base	cross-base	+Morph	+Nonce
$\mathcal{T}_{100}$	53.3	57.9 (+4.6)	59.5 (+6.2)	59.3 (+6.0)
$\mathcal{T}_{50}$	42.5	48.3 (+5.8)	49.8 (+7.3)	50.1 (+7.6)
$\mathcal{T}_{10}$	18.5	29.8 (+11.3)	34.9 (+16.4)	34.8 (+16.3)
		↓ *with fine tuning (FT)* ↓		
$\mathcal{T}_{100}$	53.3	61.3 (+8.0)	60.9 (+7.6)	**61.7 (+8.4)**
$\mathcal{T}_{50}$	42.5	52.0 (+9.5)	51.7 (+9.2)	**52.0 (+9.5)**
$\mathcal{T}_{10}$	18.5	34.7 (+16.2)	**37.3 (+18.8)**	35.4 (+16.9)

Table A.2: Effects of fine-tuning on North Sámi development data, measured in LAS. *mono-base* and *cross-base* are models without data augmentation. % improvements over *mono-base* shown in parentheses.

A.3 Cyrilic to Latin alphabet mapping

Cyrillic	Latin	Cyrillic	Latin	Cyrillic	Latin
А	A	Ф	F	қ	k
Ә	A	Х	H	л	l
Б	B	Һ	H	м	m
В	V	Ц	Ts	н	n
Г	G	Ч	Ch	ң	n
Ғ	G	Ш	Sh	о	o
Д	D	Щ	Sh	ө	o
Е	E	Ъ	’	п	p
Ё	E	Ы	Y	р	r
Ж	J	І	I	с	s
З	Z	Ь	’	т	t
И	I	Э	E	у	u
Й	I	Ю	Ju	ұ	u
К	K	Я	Ja	ү	u
Қ	K	а	a	ф	f
Л	L	ә	a	х	h
М	M	б	b	һ	h
Н	N	в	v	ц	ts
Ң	N	г	g	ч	ch
О	O	ғ	g	ш	sh
Ө	O	д	d	щ	sh
П	P	е	e	ъ	’
Р	R	ё	e	ы	y
С	S	ж	j	і	i
Т	T	з	z	ь	’
У	U	и	i	э	e
Ұ	U	й	i	ю	ju
Ү	U	к	k	я	ja

Figure A.1: The mapping from Cyrillic to Latin alphabet used in Chapter 5.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org.

Adi, Y., Kermany, E., Belinkov, Y., Lavi, O., and Goldberg, Y. (2017). Fine-grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks. *ICLR*.

Agić, Ž. (2017). Cross-Lingual Parser Selection for Low-Resource Languages. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 1–10. Association for Computational Linguistics.

Al-Rfou, R., Perozzi, B., and Skiena, S. (2013). Polyglot: Distributed word representations for multilingual nlp. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 183–192, Sofia, Bulgaria. Association for Computational Linguistics.

Alexandrescu, A. and Kirchhoff, K. (2006). Factored Neural Language Models. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 1–4, New York City, USA. Association for Computational Linguistics.

Ammar, W., Mulcaire, G., Ballesteros, M., Dyer, C., and Smith, N. (2016). Many Languages, One Parser. *Transactions of the Association for Computational Linguistics*, 4:431–444.

Anastasopoulos, A. and Chiang, D. (2018). Tied Multitask Learning for Neural Speech Translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 82–91. Association for Computational Linguistics.

Anderson, S. R. (1992). *A-Morphous Morphology*. Cambridge University Press.

Angeli, G., Johnson Premkumar, M. J., and Manning, C. D. (2015). Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, Beijing, China. Association for Computational Linguistics.

Ataman, D. and Federico, M. (2018). Compositional Representation of Morphologically-Rich Input for Neural Machine Translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 305–311. Association for Computational Linguistics.

Avramidis, E. and Koehn, P. (2008). Enriching Morphologically Poor Languages for Statistical Machine Translation. In *Proceedings of ACL-08: HLT*, pages 763–770. Association for Computational Linguistics.

Baerman, M., Brown, D., and Corbett, G. G. (2005). *The Syntax-Morphology Interface: A Study of Syncretism*. Cambridge Studies in Linguistics. Cambridge University Press.

Ballesteros, M. (2013). Effective Morphological Feature Selection with MaltOptimizer at the SPMRL 2013 Shared Task. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 63–70, Seattle, Washington, USA. Association for Computational Linguistics.

Ballesteros, M., Dyer, C., and Smith, N. A. (2015). Improved transition-based parsing by modeling characters instead of words with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 349–359, Lisbon, Portugal. Association for Computational Linguistics.

Bansal, S., Kamper, H., Livescu, K., Lopez, A., and Goldwater, S. (2019). Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume*

1 (Long and Short Papers), pages 58–68, Minneapolis, Minnesota. Association for Computational Linguistics.

Beard, R. (1988). The Separation of Derivation and Affixation: Toward a Lexeme-Morpheme Base Morphology. *Quarderni di semantica*, pages 277–287.

Belinkov, Y., Durrani, N., Dalvi, F., Sajjad, H., and Glass, J. (2017). What do Neural Machine Translation Models Learn about Morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Bender, E. M. (2013). *Linguistic Fundamentals for Natural Language Processing: 100 Essentials from Morphology and Syntax*. Morgan & Claypool Publishers.

Bergmanis, T. and Goldwater, S. (2017). From Segmentation to Analyses: a Probabilistic Model for Unsupervised Morphology Induction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 337–346, Valencia, Spain. Association for Computational Linguistics.

Bergmanis, T., Kann, K., Schütze, H., and Goldwater, S. (2017). Training Data Augmentation for Low-Resource Morphological Inflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 31–39. Association for Computational Linguistics.

Björkelund, A., Falenska, A., Yu, X., and Kuhn, J. (2017). IMS at the CoNLL 2017 UD Shared Task: CRFs and Perceptrons Meet Neural Networks. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 40–51, Vancouver, Canada. Association for Computational Linguistics.

Bohnet, B., McDonald, R., Simões, G., Andor, D., Pitler, E., and Maynez, J. (2018). Morphosyntactic Tagging with a Meta-BiLSTM Model over Context Sensitive Token Encodings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2642–2652, Melbourne, Australia. Association for Computational Linguistics.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational*

Linguistics, 5:135–146.

Botha, J. A. and Blunsom, P. (2014). Compositional Morphology for Word Representations and Language Modeling. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, Beijing, China.

Buchholz, S. and Marsi, E. (2006). CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City. Association for Computational Linguistics.

Carreras, X. and Collins, M. (2009). Non-projective Parsing for Statistical Machine Translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 200–209, Stroudsburg, PA, USA. Association for Computational Linguistics.

Caruana, R. (1997). Multitask Learning. *Machine Learning*, 28(1):41–75.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Chu, Y. J. and Liu, T. H. (1965). On the shortest arborescence of a directed graph. *Science Sinica*, 14.

Chung, J., Cho, K., and Bengio, Y. (2016). A Character-level Decoder without Explicit Segmentation for Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1693–1703, Berlin, Germany. Association for Computational Linguistics.

Coavoux, M. and Crabbé, B. (2017). Multilingual Lexicalized Constituency Parsing with Word-Level Auxiliary Tasks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 331–336. Association for Computational Linguistics.

Collobert, R. and Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the*

25th International Conference on Machine Learning, ICML '08, pages 160–167, New York, NY, USA. ACM.

Corbett, G. G. (2013). Number of Genders. In Dryer, M. S. and Haspelmath, M., editors, *The World Atlas of Language Structures Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Cotterell, R. and Heigold, G. (2017). Cross-lingual Character-Level Neural Morphological Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 748–759. Association for Computational Linguistics.

Cotterell, R., Mielke, S. J., Eisner, J., and Roark, B. (2018). Are All Languages Equally Hard to Language-Model? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 536–541, New Orleans, Louisiana. Association for Computational Linguistics.

Cotterell, R. and Schütze, H. (2015). Morphological Word-Embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1287–1292, Denver, Colorado. Association for Computational Linguistics.

Creutz, M. and Lagus, K. (2002). Unsupervised Discovery of Morphemes. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 21–30. Association for Computational Linguistics.

Creutz, M. and Lagus, K. (2007). Unsupervised Models for Morpheme Segmentation and Morphology Learning. *ACM Trans. Speech Lang. Process.*, 4(1):3:1–3:34.

Cui, H., Sun, R., Li, K., Kan, M.-Y., and Chua, T.-S. (2005). Question answering passage retrieval using dependency relations. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 400–407, New York, NY, USA. ACM.

de Lhoneux, M., Bjerva, J., Augenstein, I., and Søgaard, A. (2018). Parameter sharing between dependency parsers for related languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4992–4997. Association for Computational Linguistics.

de Lhoneux, M., Shao, Y., Basirat, A., Kiperwasser, E., Stymne, S., Goldberg, Y., and Nivre, J. (2017a). 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.*, Vancouver, Canada.

de Lhoneux, M., Stymne, S., and Nivre, J. (2017b). Arc-Hybrid Non-Projective Dependency Parsing with a Static-Dynamic Oracle. In *Proceedings of the The 15th International Conference on Parsing Technologies (IWPT).*, Pisa, Italy.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

de Marneffe, M.-C. and Manning, C. D. (2008). The Stanford Typed Dependencies Representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, Manchester, UK. Coling 2008 Organizing Committee.

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

Dozat, T., Qi, P., and Manning, C. D. (2017). Stanford's Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.

Dryer, M. S. and Haspelmath, M., editors (2013). *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Edmonds, J. (1967). Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71(4):233–240.

Fadaee, M., Bisazza, A., and Monz, C. (2017). Data Augmentation for Low-Resource Neural Machine Translation. In *Proceedings of the 55th Annual Meeting of the*

Association for Computational Linguistics (Volume 2: Short Papers), pages 567–573. Association for Computational Linguistics.

Falenska, A. and Çetinoğlu, Ö. (2017). Lexicalized vs. Delexicalized Parsing in Low-Resource Scenarios. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 18–24. Association for Computational Linguistics.

Firth, J. R. (1957). A synopsis of linguistic theory 1930-55. 1952-59:1–32.

Foley, W. (1991). *The Yimas Language of New Guinea.* Stanford University Press.

Fortescue, M. (1984). *West Greenlandic.* Croom Helm Descriptive Grammars. Croom Helm, London.

Gage, P. (1994). A new algorithm for data compression. *C Users J.*, 12(2):23–38.

Gerz, D., Vulić, I., Ponti, E. M., Naradowsky, J., Reichart, R., and Korhonen, A. (2018a). Language Modeling for Morphologically Rich Languages: Character-Aware Modeling for Word-Level Prediction. *Transactions of the Association for Computational Linguistics*, 6:451–465.

Gerz, D., Vulić, I., Ponti, E. M., Reichart, R., and Korhonen, A. (2018b). On the Relation between Linguistic Typology and (Limitations of) Multilingual Language Modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 316–327. Association for Computational Linguistics.

Godin, F., Demuynck, K., Dambre, J., De Neve, W., and Demeester, T. (2018). Explaining Character-Aware Neural Networks for Word-Level Prediction: Do They Discover Linguistic Rules? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3275–3284. Association for Computational Linguistics.

Graves, A., Fernández, S., and Schmidhuber, J. (2005). Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition. In *Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and Their Applications - Volume Part II*, ICANN'05, pages 799–804, Berlin, Heidelberg. Springer-Verlag.

Gulordava, K., Bojanowski, P., Grave, E., Linzen, T., and Baroni, M. (2018). Colorless Green Recurrent Networks Dream Hierarchically. 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 1195–1205. Association for Computational Linguistics.

Harris, Z. (1954). Distributional structure. 10.

Haspelmath, M. (2010). *Understanding Morphology*. Understanding Language Series. Arnold, London, second edition.

Heigold, G., Neumann, G., and van Genabith, J. (2017). An Extensive Empirical Evaluation of Character-Based Morphological Tagging for 14 Languages. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 505–513. Association for Computational Linguistics.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. *Master's thesis, Institut fur Informatik, Technische Universitat, Munchen.*

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780.

Howard, J. and Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339. Association for Computational Linguistics.

Hwa, R., Resnik, P., Weinberg, A., Cabezas, C., and Kolak, O. (2005). Bootstrapping parsers via syntactic projection across parallel texts. *Nat. Lang. Eng.*, 11(3):311–325.

Iggesen, O. A. (2013). Number of Cases. In Dryer, M. S. and Haspelmath, M., editors, *The World Atlas of Language Structures Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Janecki, K. (2000). *301 Polish Verbs*. 201/301 Verbs Series. Barrons Educational Series.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M., and Dean, J. (2017). Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Kann, K., Cotterell, R., and Schütze, H. (2017). One-Shot Neural Cross-Lingual Transfer for Paradigm Completion. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1993–2003. Association for Computational Linguistics.

Kann, K. and Schütze, H. (2016). *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, chapter MED: The LMU System for the SIGMORPHON 2016 Shared Task on Morphological Reinflection, pages 62–70. Association for Computational Linguistics.

Katamba, F. and Stonham, J. (2006). *Morphology: Palgrave Modern Linguistics*. Macmillan Modern Linguistics. Macmillan Education UK.

Kementchedjhieva, Y. and Lopez, A. (2018). 'Indicatements' that character language models learn English morpho-syntactic units and regularities. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 145–153, Brussels, Belgium. Association for Computational Linguistics.

Kim, Y., Jernite, Y., Sontag, D., and Rush, A. (2016). Character-Aware Neural Language Models. In *Proceedings of the 2016 Conference on Artificial Intelligence (AAAI)*.

Kiperwasser, E. and Ballesteros, M. (2018). Scheduled Multi-Task Learning: From Syntax to Translation. *Transactions of the Association for Computational Linguistics*, 6:225–240.

Kiperwasser, E. and Goldberg, Y. (2016). Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Koehn, P. and Hoang, H. (2007). Factored Translation Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Kohonen, O., Virpioja, S., and Lagus, K. (2010). Semi-Supervised Learning of Concatenative Morphology. In *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages 78–86, Uppsala, Sweden. Association for Computational Linguistics.

Kornfilt, J. (1997). *Turkish.* Descriptive Grammars. Routledge, London.

Kuhlmann, M., Gómez-Rodrìguez, C., and Satta, G. (2011). Dynamic Programming Algorithms for Transition-Based Dependency Parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 673–682. Association for Computational Linguistics.

Kuncoro, A., Ballesteros, M., Kong, L., Dyer, C., Neubig, G., and Smith, N. A. (2017). What Do Recurrent Neural Network Grammars Learn About Syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1249–1258, Valencia, Spain. Association for Computational Linguistics.

Lakretz, Y., Kruszewski, G., Desbordes, T., Hupkes, D., Dehaene, S., and Baroni, M. (2019). The emergence of number and syntax units in LSTM language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 11–20, Minneapolis, Minnesota. Association for Computational Linguistics.

Lan, W. and Xu, W. (2018). Character-Based Neural Networks for Sentence Pair Modeling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 157–163. Association for Computational Linguistics.

Landauer, T. K. and Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240.

Larasati, S. D., Kuboň, V., and Zeman, D. (2011). *Indonesian Morphology Tool (MorphInd): Towards an Indonesian Corpus*, pages 119–129. Springer Berlin Heidelberg, Berlin, Heidelberg.

Lazaridou, A., Marelli, M., Zamparelli, R., and Baroni, M. (2013). Compositionally Derived Representations of Morphologically Complex Words in Distributional Semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1517–1526, Sofia, Bulgaria. Association for Computational Linguistics.

Lee, J., Cho, K., and Hofmann, T. (2017). Fully Character-Level Neural Machine Translation without Explicit Segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378.

Lim, K., Partanen, N., and Poibeau, T. (2018). Multilingual Dependency Parsing for Low-Resource Languages: Case Studies on North Saami and Komi-Zyrian. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*. European Language Resource Association.

Ling, W., Dyer, C., Black, A. W., Trancoso, I., Fermandez, R., Amir, S., Marujo, L., and Luis, T. (2015a). Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal. Association for Computational Linguistics.

Ling, W., Trancoso, I., Dyer, C., and Black, A. W. (2015b). Character-based neural machine translation. *CoRR*, abs/1511.04586.

Linzen, T., Chrupała, G., and Alishahi, A. (2018). Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP. Brussels, Belgium. Association for Computational Linguistics.

Linzen, T., Chrupała, G., Belinkov, Y., and Hupkes, D. (2019). Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP. Florence, Italy. Association for Computational Linguistics.

Linzen, T., Dupoux, E., and Goldberg, Y. (2016). Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Luong, T., Socher, R., and Manning, C. (2013). Better Word Representations with Recursive Neural Networks for Morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria. Association for Computational Linguistics.

Makazhanov, A., Sultangazina, A., Makhambetov, O., and Yessenbayev, Z. (2015). Syntactic Annotation of Kazakh: Following the Universal Dependencies Guidelines. A report. In *3rd International Conference on Turkic Languages Processing, (TurkLang 2015)*, pages 338–350.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Comput. Linguist.*, 19(2):313–330.

Matthews, A., Neubig, G., and Dyer, C. (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. Association for Computational Linguistics.

McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 91–98, Ann Arbor, Michigan. Association for Computational Linguistics.

McDonald, R., Nivre, J., Quirmbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K., Petrov, S., Zhang, H., Täckström, O., Bedini, C., Bertomeu Castelló, N., and Lee, J. (2013). Universal Dependency Annotation for Multilingual Parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria. Association for Computational Linguistics.

McDonald, R., Petrov, S., and Hall, K. (2011). Multi-Source Transfer of Delexicalized Dependency Parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72. Association for Computational Linguistics.

McDonald, R. and Satta, G. (2007). On the Complexity of Non-Projective Data-Driven Dependency Parsing. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 121–132, Prague, Czech Republic. Association for Computational Linguistics.

Mielke, S. J. and Eisner, J. (2019). Spell once, summon anywhere: A two-level open-vocabulary language model. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 6843–6850, Honolulu.

Mikolov, T., Karafiát, M., Burget, L., Černocký, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH*

2010), volume 2010, pages 1045–1048. International Speech Communication Association.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013). Linguistic Regularities in Continuous Space Word Representations. In *HLT-NAACL*, pages 746–751.

Miyamoto, Y. and Cho, K. (2016). Gated Word-Character Recurrent Language Model. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1992–1997, Austin, Texas. Association for Computational Linguistics.

Nguyen, D.-H. (1997). *Vietnamese*, volume 9 of *London Oriental and African Language Library*. John Benjamins, Amsterdam.

Nivre, J. (2009). Non-Projective Dependency Parsing in Expected Linear Time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 351–359. Association for Computational Linguistics.

Nivre, J., Abrams, M., Agić, Ž., Ahrenberg, L., Antonsen, L., Aranzabe, M. J., Arutie, G., Asahara, M., Ateyah, L., Attia, M., Atutxa, A., Augustinus, L., Badmaeva, E., Ballesteros, M., Banerjee, E., Bank, S., Barbu Mititelu, V., Bauer, J., Bellato, S., Bengoetxea, K., Bhat, R. A., Biagetti, E., Bick, E., Blokland, R., Bobicev, V., Börstell, C., Bosco, C., Bouma, G., Bowman, S., Boyd, A., Burchardt, A., Candito, M., Caron, B., Caron, G., Cebiroğlu Eryiğit, G., Celano, G. G. A., Cetin, S., Chalub, F., Choi, J., Cho, Y., Chun, J., Cinková, S., Collomb, A., Çöltekin, Ç., Connor, M., Courtin, M., Davidson, E., de Marneffe, M.-C., de Paiva, V., Diaz de Ilarraza, A., Dickerson, C., Dirix, P., Dobrovoljc, K., Dozat, T., Droganova, K., Dwivedi, P., Eli, M., Elkahky, A., Ephrem, B., Erjavec, T., Etienne, A., Farkas, R., Fernandez Alcalde, H., Foster, J., Freitas, C., Gajdošová, K., Galbraith, D., Garcia, M., Gärdenfors, M., Gerdes, K., Ginter, F., Goenaga, I., Gojenola, K., Gökırmak, M., Goldberg, Y., Gómez Guinovart, X., Gonzáles Saavedra, B., Grioni, M., Grūzītis, N., Guillaume, B., Guillot-Barbance, C., Habash, N., Hajič, J., Hajič jr., J., Hà Mỹ, L., Han, N.-R., Harris, K., Haug, D., Hladká, B., Hlaváčová, J., Hociung, F., Hohle, P., Hwang, J., Ion, R., Irimia, E., Jelínek, T., Johannsen, A., Jørgensen, F., Kaşıkara, H., Kahane, S., Kanayama, H., Kanerva, J., Kayadelen, T., Kettnerová, V., Kirchner, J., Kotsyba, N., Krek, S., Kwak, S., Laippala, V., Lambertino, L., Lando, T., Larasati, S. D., Lavrentiev, A., Lee, J., Lê Hồng, P., Lenci, A., Lertpradit, S., Leung, H., Li, C. Y., Li, J., Li, K., Lim, K., Ljubešić, N., Loginova, O., Lyashevskaya, O., Lynn, T., Macketanz, V., Makazhanov, A., Mandl, M., Manning, C., Manurung, R., Mărănduc, C., Mareček, D., Marheinecke, K., Martínez Alonso, H., Martins, A., Mašek, J., Matsumoto, Y., McDonald, R., Mendonça, G., Miekka, N., Missilä, A., Mititelu, C., Miyao, Y., Montemagni, S., More, A., Moreno Romero, L., Mori, S., Mortensen, B., Moskalevskyi, B., Muischnek, K., Murawaki, Y., Müürisep, K., Nainwani, P., Navarro Horñiacek, J. I., Nedoluzhko, A., Nešpore-Bērzkalne, G., Nguyễn Thị, L., Nguyễn Thị Minh, H., Nikolaev, V., Nitisaroj, R., Nurmi, H., Ojala, S., Olúòkun, A., Omura, M., Osenova, P., Östling, R., Øvrelid, L., Partanen, N., Pascual, E., Passarotti, M., Patejuk, A., Peng, S., Perez, C.-A., Perrier, G., Petrov, S., Piitulainen, J., Pitler, E., Plank, B., Poibeau, T., Popel, M., Pretkalniņa, L., Prévost, S., Prokopidis, P., Przepiórkowski, A., Puolakainen, T., Pyysalo, S., Rääbis, A., Rademaker, A., Ramasamy, L., Rama, T., Ramisch, C., Ravishankar, V., Real, L., Reddy, S., Rehm, G., Rießler, M., Rinaldi, L., Rituma, L., Rocha, L., Romanenko, M., Rosa, R., Rovati, D., Roca, V., Rudina, O., Sadde, S., Saleh, S., Samardžić, T., Samson, S., Sanguinetti, M., Saulīte, B., Sawanakunanon, Y., Schneider, N., Schuster, S., Seddah, D., Seeker, W., Seraji, M., Shen, M., Shimada, A., Shohibussirri, M., Sichinava, D., Silveira, N., Simi, M., Simionescu, R., Simkó, K., Šimková, M., Simov, K., Smith, A., Soares-Bastos, I., Stella, A., Straka, M., Strnadová, J., Suhr, A., Sulubacak, U., Szántó, Z., Taji, D., Takahashi, Y., Tanaka, T., Tellier, I., Trosterud, T., Trukhina, A., Tsarfaty, R., Tyers, F., Uematsu, S., Urešová, Z., Uria, L., Uszkoreit, H., Vajjala, S., van Niekerk, D., van Noord, G., Varga, V., Vincze, V., Wallin, L., Washington, J. N., Williams, S., Wirén, M., Woldemariam, T., Wong, T.-s., Yan, C., Yavrumyan, M. M., Yu, Z., Žabokrtský, Z., Zeldes, A., Zeman, D., Zhang, M., and Zhu, H. (2018). Universal Dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Nivre, J., Agić, Ž., Aranzabe, M. J., Asahara, M., Atutxa, A., Ballesteros, M., Bauer, J., Bengoetxea, K., Bhat, R. A., Bosco, C., Bowman, S., Celano, G. G. A., Connor, M., de Marneffe, M.-C., Diaz de Ilarraza, A., Dobrovoljc, K., Dozat, T., Erjavec, T., Farkas, R., Foster, J., Galbraith, D., Ginter, F., Goenaga, I., Gojenola, K., Goldberg, Y., Gonzales, B., Guillaume, B., Hajič, J., Haug, D., Ion, R., Irimia, E., Johannsen, A., Kanayama, H., Kanerva, J., Krek, S., Laippala, V., Lenci, A., Ljubešić, N., Lynn, T., Manning, C., Mrnduc, C., Mareček, D., Martínez Alonso, H., Mašek, J., Matsumoto, Y., McDonald, R., Missilä, A., Mititelu, V., Miyao, Y., Montemagni, S., Mori, S., Nurmi, H., Osenova, P., Øvrelid, L., Pascual, E., Passarotti, M., Perez, C.-A., Petrov, S., Piitulainen, J., Plank, B., Popel, M., Prokopidis, P., Pyysalo, S., Ramasamy, L., Rosa, R., Saleh, S., Schuster, S., Seeker, W., Seraji, M., Silveira, N., Simi, M., Simionescu, R., Simkó, K., Simov, K., Smith, A., Štěpánek, J., Suhr, A., Szántó, Z., Tanaka, T., Tsarfaty, R., Uematsu, S., Uria, L., Varga, V., Vincze, V., Žabokrtský, Z., Zeman, D., and Zhu, H. (2015). Universal Dependencies 1.2. LINDAT/CLARIN digital library at Institute of Formal and Applied Linguistics, Charles University in Prague.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal Dependencies v1: A Multilingual Treebank Collection. In Chair), N. C. C., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

Nivre, J. et al. (2017). Universal Dependencies 2.0. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague, http://hdl.handle.net/11234/1-1983.

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., and Yuret, D. (2007). The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 915–932, Prague, Czech Republic. Association for Computational Linguistics.

Pasha, A., Al-Badrashiny, M., Diab, M., Kholy, A. E., Eskander, R., Habash, N., Pooleery, M., Rambow, O., and Roth, R. (2014). MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. In Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1094–1101, Reykjavik, Iceland. European Language Resources Association (ELRA). ACL Anthology Identifier: L14-1479.

Patejuk, A. and Przepiórkowski, A. (2014). Structural Case Assignment to Objects in Polish. In Butt, M. and King, T. H., editors, *The Proceedings of the LFG'14 Conference*, pages 429–447, Stanford, CA. CSLI Publications.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proc. of NAACL*.

Pinter, Y., Marone, M., and Eisenstein, J. (2019). Character Eyes: Seeing Language through Character-Level Taggers. In *Proceedings of ACL workshop on Analyzing and interpreting neural networks for NLP (BlackBoxNLP)*.

Qiu, S., Cui, Q., Bian, J., Gao, B., and Liu, T.-Y. (2014). Co-learning of Word Representations and Morpheme Representations. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 141–150, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Rademaker, A., Chalub, F., Real, L., Freitas, C., Bick, E., and de Paiva, V. (2017). Universal Dependencies for Portuguese. In *Proceedings of the Fourth International Conference on Dependency Linguistics (Depling)*, pages 197–206, Pisa, Italy.

Rasooli, M. S. and Collins, M. (2017). Cross-Lingual Syntactic Transfer with Limited Resources. *Transactions of the Association for Computational Linguistics*, 5:279–293.

Rei, M., Crichton, G., and Pyysalo, S. (2016). Attending to Characters in Neural Sequence Labeling Models. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 309–318, Osaka, Japan. The COLING 2016 Organizing Committee.

Roark, B. and Sproat, R. (2007). *Computational Approach to Morphology and Syntax*. Oxford University Press.

Rosa, R. and Mareček, D. (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*, pages 187–196. Association for Computational Linguistics.

Rubino, C. (2013). Reduplication. In Dryer, M. S. and Haspelmath, M., editors, *The World Atlas of Language Structures Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Sahin, G. G. and Steedman, M. (2018a). Character-Level Models versus Morphology in Semantic Role Labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 386–396. Association for Computational Linguistics.

Sahin, G. G. and Steedman, M. (2018b). Data Augmentation via Dependency Tree Morphing for Low-Resource Languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5004–5009. Association for Computational Linguistics.

Sahin, G. G., Vania, C., Kuznetsov, I., and Gurevych, I. (2019). LINSPECTOR: multilingual probing tasks for word representations. *CoRR*, abs/1903.09442.

Santos, C. D. and Zadrozny, B. (2014). Learning Character-level Representations for Part-of-Speech Tagging. In Xing, E. P. and Jebara, T., editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1818–1826, Bejing, China. PMLR.

Schuster, M. and Nakajima, K. (2012). Japanese and Korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Seddah, D., Kübler, S., and Tsarfaty, R. (2014). Introducing the SPMRL 2014 Shared Task on Parsing Morphologically-rich Languages. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 103–109, Dublin, Ireland. Dublin City University.

Seddah, D., Tsarfaty, R., Kübler, S., Candito, M., Choi, J. D., Farkas, R., Foster, J., Goenaga, I., Gojenola Galletebeitia, K., Goldberg, Y., Green, S., Habash, N., Kuhlmann, M., Maier, W., Nivre, J., Przepiórkowski, A., Roth, R., Seeker, W., Versley, Y., Vincze, V., Woliński, M., Wróblewska, A., and Villemonte de la Clergerie, E. (2013). Overview of the SPMRL 2013 Shared Task: A Cross-Framework Evaluation of Parsing Morphologically Rich Languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 146–182, Seattle, Washington, USA. Association for Computational Linguistics.

Seeker, W. and Kuhn, J. (2013). Morphological and Syntactic Case in Statistical Dependency Parsing. *Computational Linguistics*, 39(1):23–55.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. Association for Computational Linguistics.

Sheyanova, M. and Tyers, F. M. (2017). Annotation schemes in North Sámi dependency parsing. In *Proceedings of the 3rd International Workshop for Computational Linguistics of Uralic Languages*, pages 66–75.

Shi, T., Wu, F. G., Chen, X., and Cheng, Y. (2017). Combining Global Models for Parsing Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 31–39, Vancouver, Canada. Association for Computational Linguistics.

Shi, X., Padhi, I., and Knight, K. (2016). Does String-Based Neural MT Learn Source Syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534. Association for Computational Linguistics.

Smit, P., Virpioja, S., Grönroos, S.-A., and Kurimo, M. (2014). Morfessor 2.0: Toolkit for statistical morphological segmentation. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 21–24, Gothenburg, Sweden. Association for Computational Linguistics.

Smith, A., Bohnet, B., de Lhoneux, M., Nivre, J., Shao, Y., and Stymne, S. (2018a). 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*, pages 113–123. Association for Computational Linguistics.

Smith, A., de Lhoneux, M., Stymne, S., and Nivre, J. (2018b). An Investigation of the Interactions Between Pre-Trained Word Embeddings, Character Models and POS Tags in Dependency Parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2711–2720, Brussels, Belgium. Association for Computational Linguistics.

Socher, R., Lin, C. C.-Y., Ng, A. Y., and Manning, C. D. (2011). Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of* *the 28th International Conference on International Conference on Machine Learning*, page 129136, Madison, WI, USA. Omnipress.

Søgaard, A. (2011). Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 682–686. Association for Computational Linguistics.

Søgaard, A. and Goldberg, Y. (2016). Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235. Association for Computational Linguistics.

Sperr, H., Niehues, J., and Waibel, A. (2013). Letter N-Gram-based Input Encoding for Continuous Space Language Models. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pages 30–39, Sofia, Bulgaria. Association for Computational Linguistics.

Straka, M. (2018). UDPipe 2.0 Prototype at CoNLL 2018 UD Shared Task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207. Association for Computational Linguistics.

Straka, M. and Straková, J. (2017). Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

Stymne, S., de Lhoneux, M., Smith, A., and Nivre, J. (2018). Parser Training with Heterogeneous Treebanks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 619–625. Association for Computational Linguistics.

Sylak-Glassman, J. (2016). The composition and use of the universal morphological feature schema (unimorph schema). Technical report.

Tiedemann, J. and Agic, v. (2016). Synthetic treebanking for cross-lingual dependency parsing. *J. Artif. Int. Res.*, 55(1):209–248.

Tokui, S., Oono, K., Hido, S., and Clayton, J. (2015). Chainer: a Next-Generation Open Source Framework for Deep Learning. In *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)*.

Tsarfaty, R., Seddah, D., Goldberg, Y., Kuebler, S., Versley, Y., Candito, M., Foster, J., Rehbein, I., and Tounsi, L. (2010). Statistical Parsing of Morphologically Rich Languages (SPMRL) What, How and Whither. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 1–12. Association for Computational Linguistics.

Tsarfaty, R. and Sima'an, K. (2008). Relational-realizational parsing. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 889–896, Manchester, UK. Coling 2008 Organizing Committee.

Tsvetkov, Y., Sitaram, S., Faruqui, M., Lample, G., Littell, P., Mortensen, D., Black, A. W., Levin, L., and Dyer, C. (2016). Polyglot Neural Language Models: A Case Study in Cross-Lingual Phonetic Representation Learning. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1357–1366. Association for Computational Linguistics.

Turney, P. D. and Pantel, P. (2010). From Frequency to Meaning: Vector Space Models of Semantics. *J. Artif. Int. Res.*, 37(1):141–188.

Tyers, F. M. and Washington, J. N. (2015). Towards a Free/Open-source Universal-dependency Treebank for Kazakh. In *3rd International Conference on Turkic Languages Processing, (TurkLang 2015)*, pages 276–289.

Vania, C., Grivas, A., and Lopez, A. (2018). What do character-level models learn about morphology? The case of dependency parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2573–2583. Association for Computational Linguistics.

Vania, C. and Lopez, A. (2017). From Characters to Words to in Between: Do We Capture Morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2016–2027. Association for Computational Linguistics.

Vania, C. and Lopez, A. (2018). Explicitly modeling case improves neural dependency parsing. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 356–358, Brussels, Belgium. Association for Computational Linguistics.

Veldhoen, S., Hupkes, D., and Zuidema, W. (2016a). Diagnostic Classifiers: Revealing how Neural Networks Process Hierarchical Structure. In *Pre-Proceedings of the Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches (CoCo @ NIPS 2016).*

Veldhoen, S., Hupkes, D., and Zuidema, W. H. (2016b). Diagnostic Classifiers Revealing how Neural Networks Process Hierarchical Structure. In *CoCo@NIPS*.

Virpioja, S., Turunen, V. T., Spiegler, S., Kohonen, O., and Kurimo, M. (2011). Empirical Comparison of Evaluation Methods for Unsupervised Learning of Morphology. *Traitement Automatique des Langues*, 52(2):45–90.

Vylomova, E., Cohn, T., He, X., and Haffari, G. (2017). Word Representation Models for Morphologically Rich Languages in Neural Machine Translation. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 103–108. Association for Computational Linguistics.

Wang, D. and Eisner, J. (2018). Synthetic Data Made to Order: The Case of Parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1325–1337. Association for Computational Linguistics.

Wieting, J., Bansal, M., Gimpel, K., and Livescu, K. (2016). Charagram: Embedding words and sentences via character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1504–1515, Austin, Texas. Association for Computational Linguistics.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G. S., Hughes, M., and Dean, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *ArXiv*, abs/1609.08144.

Zampieri, M., Malmasi, S., Ljubešić, N., Nakov, P., Ali, A., Tiedemann, J., Scherrer, Y., and Aepli, N. (2017). Findings of the VarDial Evaluation Campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 1–15. Association for Computational Linguistics.

Zeman, D., Haji{č}, J., Popel, M., Potthast, M., Straka, M., Ginter, F., Nivre, J., and Petrov, S. (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. Association for Computational Linguistics.

Zeman, D., Popel, M., Straka, M., Hajic, J., Nivre, J., Ginter, F., Luotolahti, J., Pyysalo, S., Petrov, S., Potthast, M., Tyers, F., Badmaeva, E., Gokirmak, M., Nedoluzhko, A., Cinkova, S., Hajic jr., J., Hlavacova, J., Kettnerová, V., Uresova, Z., Kanerva, J., Ojala, S., Missilä, A., Manning, C. D., Schuster, S., Reddy, S., Taji, D., Habash, N., Leung, H., de Marneffe, M.-C., Sanguinetti, M., Simi, M., Kanayama, H., dePaiva, V., Droganova, K., Martínez Alonso, H., Çöltekin, c., Sulubacak, U., Uszkoreit, H., Macketanz, V., Burchardt, A., Harris, K., Marheinecke, K., Rehm, G., Kayadelen, T., Attia, M., Elkahky, A., Yu, Z., Pitler, E., Lertpradit, S., Mandl, M., Kirchner, J., Alcalde, H. F., Strnadová, J., Banerjee, E., Manurung, R., Stella, A., Shimada, A., Kwak, S., Mendonca, G., Lando, T., Nitisaroj, R., and Li, J. (2017a). CoNLL 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

Zeman, D., Popel, M., Straka, M., Hajic, J., Nivre, J., Ginter, F., Luotolahti, J., Pyysalo, S., Petrov, S., Potthast, M., Tyers, F., Badmaeva, E., Gokirmak, M., Nedoluzhko, A., Cinkova, S., Hajic jr., J., Hlavacova, J., Kettnerová, V., Uresova, Z., Kanerva, J., Ojala, S., Missilä, A., Manning, C. D., Schuster, S., Reddy, S., Taji, D., Habash, N., Leung, H., de Marneffe, M.-C., Sanguinetti, M., Simi, M., Kanayama, H., dePaiva, V., Droganova, K., Martínez Alonso, H., Çöltekin, Ç., Sulubacak, U., Uszkoreit, H., Macketanz, V., Burchardt, A., Harris, K., Marheinecke, K., Rehm, G., Kayadelen, T., Attia, M., Elkahky, A., Yu, Z., Pitler, E., Lertpradit, S., Mandl, M., Kirchner, J., Alcalde, H. F., Strnadová, J., Banerjee, E., Manurung, R., Stella, A., Shimada, A., Kwak, S., Mendonca, G., Lando, T., Nitisaroj, R., and Li, J. (2017b). CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In

Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pages 1–19. Association for Computational Linguistics.

Zeman, D. and Resnik, P. (2008). Cross-Language Parser Adaptation between Related Languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*.

Zhang, X., Cheng, J., and Lapata, M. (2017). Dependency Parsing as Head Selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 665–676, Valencia, Spain. Association for Computational Linguistics.

Zhou, C. and Neubig, G. (2017). Multi-space Variational Encoder-Decoders for Semi-supervised Labeled Sequence Transduction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 310–320. Association for Computational Linguistics.