Computational Morphology in Low-Resource Settings

Synthesis Paper

Sarah Moeller

Contents

1 Introduction

This paper provides an overview of common natural language processing (NLP) methods and how those methods have been applied to the study of morphology, particularly in low-resource languages (LRL). For NLP, work with LRL is still uncharted territory. This paper explores the limited work in morphology by adapting this question: "What [computational] methods...can detect [morphological] structure in small, noisy data sets, while being directly applicable to a wide variety of languages?" (Bird, 2009). This question will guide this paper through its ultimate purpose of exploring the possibilities of training computational morphological models on data produced by language documentation and description (LDD) field projects.[1] Therefore, throughout the paper models or techniques are identified that seem likely to be successful if integrated into linguistic fieldwork.

Morphology comprises word-building properties in human languages and their accompanying (morpho-)syntactic phenomena. Historically, computational linguists and "paper-and-pencil linguists" have taken different and sometimes seemingly incompatible approaches to morphology (Karttunen and Beesley, 2005). Yet, despite their out-of-sync approaches, both computational linguistics and "traditional" linguistics benefit from morphological analysis (Cotterell et al., 2015, p. 165).

In general linguistics, morphological description of a broad range of languages is a foundational step towards any reasonable linguistic theory. Therefore, analyzing the morphological system is an early step in linguistic study. Still, the field of linguistics is not interested solely in supporting theory. Early linguists focused on increasing human understanding of language by describing a language's structure, including morphology, based on documented data. This focus has been revived since the 1990's with the establishment of language documentation as a distinct subfield. This

[1] In this paper, LDD data refers to resources that results from linguistic fieldwork and basic linguistic analysis. According to the description of LORELEI (a US-government funded project for building language technology in low-resource languages), LRL refers to languages for which no automated human language technology exists, typically because of a lack of available linguistic resources. Other terms are sometimes used in this paper: "under-described languages" are languages with minimal published descriptive linguistic resources, what has been called "very scarce-resource language" (Duong, 2017); whereas "under-documented languages", or Duong's "extremely scarce-resource languages," lack sufficient raw or annotated data to write a full descriptive grammar; "endangered languages" are predicted to have no native speakers within a generation or two and most are under-documented and/or under-described. These distinctions are rarely crucial in this paper and can be understood almost interchangeably.

subfield also holds interest in practical downstream goals such as language technology, language conservation, and language learning.[2]

In NLP, computational morphology attempts to parse or generate valid inflected forms in order to improve downstream NLP tasks such as machine translation or voice recognition. Among some computational linguists there seems to be a belief that linguistic theory has little benefit for computational methods (Goldsmith et al., 2017). Hypothesizing an all-encompassing theory of language does not induce a workable NLP tool from data. It has even been questioned whether computational linguistics needs morphological analysis at all, particularly with the recent trend of end-to-end deep learning which has the potential to improve downstream tasks without explicitly modeling morphology. Might NLP goals benefit equally well by representing words as strings of characters? Although character-level models can learn relationships between orthographically similar words, comparing results on ten languages showed that such models are too easily led up the garden path by orthographic signals (Vania and Lopez, 2017). Language models trained on morphological patterns provide greater predictive accuracy.

Morphological analysis is particularly important when working with morphologically complex languages. Languages that build words from multiple morphemes or via significant morphophonological changes produce a high number of inflected and compound words which appear to the machine as brand new, unrelated words (Dreyer and Eisner, 2011; Goldsmith et al., 2017; Hammarström and Borin, 2011; Kann et al., 2016; Ruokolainen et al., 2013). These include agglutinating morphologies (common in central and north Asia, South America, central and southern Africa, Australia), polysynthetic (North America, the Far East of Russia), or non-concatenative (north Africa and the Middle East, southeast Asia). The fact that computational linguists ask whether morphological analysis is even necessary hints at the out-sized role that high resource languages have played in natural language processing. The most common languages used in computational linguistics are official languages in European countries, Chinese, and Arabic. Most of these languages have comparatively simple fusional or isolating morphology, with a very few (e.g. Finnish)

[2] A casual perusal of the flagship journal *Language Documentation and Conservation* journal attests this interest.

that belong to the somewhat more complicated agglutinative or non-concatenative types (i.e. Arabic). None belong to the much more complicated polysynthetic type, none are under-documented, none are endangered languages. While the field of linguistics has slowly but surely widened the world's knowledge about human language, thanks in part to the recent emphasis on documentary fieldwork, computational linguistics has yet to truly expand beyond a handful of economically or politically powerful languages. As an example, when Vania and Lopez (2017) compared morpheme-level versus character-level representation, they selected ten languages that do in fact represent a range of morphological phenomena, but still still only represent six language families. Three are Indo-European languages characterized by fusional morphology (though English tends toward the even simpler isolating type). One (Finnish) is a European, but not Indo-European, language with agglutinative morphology and many other interesting morphological and phonological alternations. Four others (Japanese, Turkish, Indonesian and Malaysian) are also agglutinative. The other two languages (Hebrew and Arabic) are Semitic languages, famous for the templatic type of non-concatenative morphology. Even this narrow dataset demonstrated that the more complex a language's morphology is the more crucial the role that morphological analysis plays in NLP performance.

The rest of this paper assumes that morphology does have a role to play in both computational and general linguistics. Section 2 defines and compares the tasks of morphology learning in computational linguistics and language documentation and description. This paper also assumes that NLP algorithms and methods should be expanded into more low-resource languages. Section 3 reviews the literature on various computational approaches to morphology, how they have been applied to low-resource languages, and what their advantages and disadvantages are in low-resource settings.

Two issues arise throughout the paper. First, how can machine learning be improved when data is limited? Second, what are the most promising computational methods for LDD? In addition to the reviews in section 3, section 4 presenting specific techniques that have overcome data limitations. Section 5 addresses the second question but goes beyond specific methods by envisioning how

computational linguistics and language documentation and description could together address the issue of limited data which plagues both fields. It explains a bottleneck in language documentation and description and suggests how the models and technique described in this paper could address it.

2 Morphology

This section describes morphology and the questions it studies. It also contrasts how computational linguistics and general linguistics approach morphology. The two subsections provide a more detailed look at the two parts of morphology that the rest of the paper will focus on.

The study of morphology is the inference of rules that govern a language's word building strategies, including morphophonological changes in morpheme shapes (pronunciation) and the discovery of systematically related word forms (derivational morphology and inflectional paradigms) (Roark and Sproat, 2007). According to Goldsmith et al. (2017), an ideal morphological model answers the following questions, among others:

- What are the morphemes in each word? What are their meanings or functions?
- What morphological paradigms exist in the language? What morphological features distinguish them from one another?
- What allomorphs, or alternative pronunciations, does each morpheme have? Under what conditions do the allomorphs appear?
- What combinations of morphemes does the language permit on each lexical category (part of speech (POS))?
- What are the morphological processes that significantly change a word's meaning or part of speech? How productive are these derivational processes?

Such a complete morphological model is difficult to accomplish and even more difficult to assess. A computational model would need to be computable, robust, and interpretable, while accounting for all the language's morphophonology, allomorphy, and for any ambiguous inflected forms (Virpioja et al., 2011). In other words, an ideal computational model would be nearly as good as human linguists, who are also able to account for the ever-present inconsistencies and

unsystematic exceptions that occur in natural languages. While descriptive linguists attempt to answer all the above questions (and more), most effort in computational morphology has been expended on the first questions, with some significant effort given to the second set.

Attempts to answer the first set of questions about the number, shape, and meaning of morphemes is the core of morphological analysis. Traditionally, morphemes are first identified, for example, *-ed* would be recognized as a segment with its own minimal meaning. The meanings, or functions, that each morpheme contributes must be deduced. For example, *-ed* can mean 'PAST' or 'PARTICIPLE'. After the analysis of morphemes, it becomes possible to delve deeper into a language's morphology, asking questions about morphological patterns.

Patterns that govern how morphemes combine on a word and which morphemes can or cannot co-occur are inferred from morphologically analyzed and annotated data. Once the morpheme is identified in natural occurring texts, it should be clear, for example, that English uses *-ed* as a suffix to add past tense or participial meaning to a verb stem. Sets of rules may govern the morphological patterns of certain classes of morphemes and these patterns are exemplified as paradigms such as shown in Table 1. Thus, for example, the class of "regular" English uses the suffix *-ed* (versus "irregular" *swim, swam, swum*, etc.).

Before continuing, it is worth emphasizing a subtle but significant difference between computational and general linguistics that was mentioned in section 1. Linguistics explores abstract forms and builds theories. Computers do not deal in abstractions; they handle what they "see". Computational linguistics deals primarily with textual representations, whether orthographic or phonetic. For example, the English past tense morpheme can be pronounced in three ways: [d], [t], [ɨd]. The orthographic representation *ed* is an abstraction to indicate the three allomorphs have a unified meaning/function. If the texts are represented orthographically, then the suffix *-ed* refers to the abstract form. If it is represented with phonetically, then each of the allomorphs will be dealt with separately by a computation model that is not tuned to the textual representation.

2.1 Morphological Analysis and Annotation

Morphological analysis can be separated into two core tasks (Cotterell et al., 2015; Hammarström and Borin, 2011; Nicolai and Kondrak, 2017; Palmer, 2009). The first task is identifying morphemes by determining their shapes and marking boundaries between them, as done in (1b). This task is called (unlabeled) morpheme segmentation (Creutz and Lagus, 2007b; Snyder and Barzilay, 2008). The second task is deducing each morpheme's meaning. This is known as labeled morpheme segmentation, parsing, or glossing. When parsing follows segmentation, it means associating each morpheme with a label (gloss) that indicates its meaning or morphosyntactic function, as done in (1c). These two tasks are a large part of linguistic data annotation. When performed computationally, both tasks are more properly seen as annotation because a computer algorithm cannot make analytical decisions (though it finds complicated patterns and makes probablistic predictions that may seem like analytical decisions).

(1) a. walked

b. walk-ed

c. perambulate-PAST

d. walk [PAST]

Parsing—discovering morphemes' meanings—does not necessarily require morpheme identification and segmentation. It is possible to parse a word without identifying the individual morphemes. In other words, it is possible to produce (1c) without first producing (1b). Parsing without segmentation is more common in computational methods. Computational morphology aims to take texts from any human language and build a model that accounts for every word in them, then generalize the model to unseen words in new texts (Goldsmith et al., 2017). If the immediate task can be accomplished without morpheme identification and segmentation, a model may skip it, as, for example, in one CoNLL-SIGMORPHON 2018 shared task, shown in (2), where words were inflected without identifying the language's morphemes (Cotterell et al., 2018). Parsing may also

include a lemmatization step, where the orthographic representation of the stem morpheme is identified, as in (1d). This is a standard approach of hand-constructed morphological analyzers (see section 3.1) which output the lemma along with a bundle of tags that indicate the morphosyntactic features of the inflected form.

(2) a. **INPUT:** The dogs are _____ . *walking*

b. **OUTPUT:** walking

Nicolai and Kondrak (2017) divide the morphological analysis subtasks slightly differently, making a distinction between morphological "analysis" and morphological tagging. They describe morphological analysis as a combination of segmentation and labeling, that is (1b) and (1c) together, though they later state that "morphological tagging can be performed as a downstream application of morphological analysis". They thereby adhere to the same two distinctions described above. Virpioja et al. (2011) adds a third task: identification of morphologically related words. However, this is more properly seen as a later step in general morphological description, and a first step to identifying inflectional classes and the systematic rules that govern them (see section 2.2).

Computationally, morpheme segmentation algorithms are quite similar to word segmentation approaches. They divides strings of text. This is sometimes called surface segmentation (e.g. "runn-ing") in contrast to canonical segmentation where abstract forms are provided (e.g. "run-ing"). Computational segmentation tends to perform better on concatenative morphology, where words are built from morphemes like beads on a string such as in example **??**. Computational modeling of non-concatenative morphology, particularly when it occurs in a language that combines with concatenative morphology, remains "arguably one of the main current challenges of the field" and attempts "have been mostly applied to Arabic" (Goldsmith et al., 2017).

Computational approaches can be lexicon-based, focusing on detecting morpheme shapes, or it can focus on detecting morpheme boundaries (Goodman, 2013). Most lexicon-based approaches learn a generative model. They create a model of word forms and can generate them. Most bound-

ary detection models perform discriminative learning, probabilistically estimating the segmentation boundaries in a given word. Both provide a lexicon of morphemes which is more useful than a vocabulary of inflected words as they appear in running text (Creutz and Lagus, 2002).

During morpheme segmentation, the machine is not usually tasked to identify allomorphy, though some learning of allomorphy and morphophonogical changes may happen before this step (Goldsmith et al., 2017). It is done without regard to the words' potential relation (Virpioja et al., 2011). Ideally, parsing accounts for allomorphy and other complicated morphophonogical issues. This is difficult to do without some previous analysis so it is no surprise that the most accurate, and historically most popular, computational morphological models are finite state transducers (FSTs). FSTs can take advantage of published linguistic descriptions. A morpheme lexicon and collection of a language's morphophonological rules are manually constructed (Beesley and Karttunen, 2003).

In descriptive linguistics, the two tasks are not distinguished because they are typically tackled simultaneously. It is more likely that linguistics would categorize morphological analysis as all the tasks leading to the identification of morphemes, their meanings, and the description of systematic rules and relationships between words. The mechanical parts of these tasks—segmenting and glossing morphemes, together with rough translations of sentences, is called interlinearization, as shown in Figure 1 (see section 5).

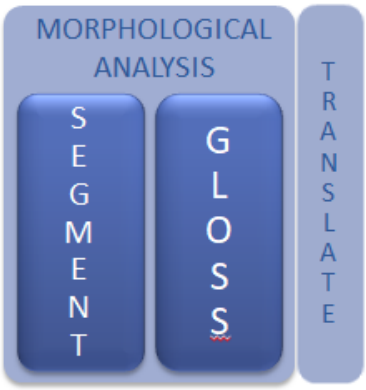


Figure 1: Tasks of interlinearization. Words are segmented and morpheme boundaries are marked. Each morpheme is glossed (parsed). Finally, each sentence is translated in a language of wider communication to clarify how the morphemes compose meaning.

In computational morphology, the two tasks, if both are attempted, are tackled sequentially. First, the machine finds morpheme boundaries, then attaches labels. However, some computational models have taken a tip from the LDD workflow and joined the two tasks. In some cases this yields higher accuracy. It also allows the machine to go beyond annotation and "learn a probabilistic model of morphotactics [rules about the ordering and co-occurrence of morphemes on a word]" (Cotterell et al., 2015, p. 165).

Morphological segmentation and parsing should be distinguished from syntactic annotation, of which the most common type is part-of-speech (POS) tagging. However, this distinction is problematic. Some POS tag sets, such as the expanded set of Universal Dependency tags (De Marneffe et al., 2014), include morphologically-marked syntactic features like number or grammatical case. In fact, "morphological" tag sets and "morphological" annotation could benefit from even more syntactic information (Cotterell and Heigold, 2017).

LDD distinguishes morphological annotation from syntactic annotation even less strictly than computational linguistics. IGTs often include lexical categories (POS) and information related to syntax, phonology, and even non-verbal communication. The fact is that the divisions (morphology, syntax, phonology, etc.) are convenient for linguistic science but are not as clear in natural language. The division between morphology and syntax is more obvious in the relatively simple morphological structures that many high resource languages have, such as the fusional morphology of most European languages or the isolating morphology of Chinese languages. Both fusional and isolating morphologies use independent function words (e.g. prepositions, particles, etc.) rather than bound morphemes to express a significant amount of syntactic information. Even some more complex morphological structures, such as the non-concatenative, templatic Arabic, also expresses a fair amount of syntactic information with independent words. These strategies stand in contrast to agglutinative and polysynthetic morphology characterizing many low-resource languages, which express most or all syntactic information by bound morphemes on the inflected word forms.

2.2 Morphological Paradigm Induction

The study of a language's morphology extends to the inference of how morphemes compose related words and interact with each other. If this is limited to how morphemes change a lexeme's form in order to add or alter grammatical meaning, then it is called inflectional morphology. The inference of inflectional rules is based on three assumptions (Durrett and DeNero, 2013). First, within each lexical category, the patterns of inflection are dictated by its own subsystem of rules. Russian nouns, for example, can be generalized into three simplified patterns of inflection, shown in Table 1. Lexemes that follow the same pattern belong to the same inflectional class (sometimes called "declensions" for nouns and adjectives and "conjugations" for verbs). The patterns are sometimes called inflectional paradigms. Second, inflectional patterns are triggered by context and, therefore, the morphological rules dictating the patterns can be inferred from context. Descriptive studies look to phonology for the triggering context or else to both phonological structure and the semantic content of the lexeme. Computational models look to orthographic context. The third assumption is that every stem morpheme is inflected consistently. The verb "walk" will always be inflected as a regular verb, never as a irregular verb.

A lexeme's inflectional class becomes a useful part of its dictionary entry. The class's paradigm is described in table which displays either just the inflectional affix(es), as in Table 1, or a sample lexeme with all its inflected forms. Paradigm tables are featured in published grammatical descriptions and language learning material.

	Class 1		**Class 2**		**Class 3**	
	SG	PL	SG	PL	SG	PL
NOM	-a	-i	Ø	-i	-j	-i
ACC	-u	-i/Ø	Ø/-a	-i/-ov	-j	-i/-jej
GEN	-i	Ø	-a	-ov	-i	-jej
DAT	-je	-am	-u	-am	-i	-jam
INST	-oj	-ami	-om	-ami	-ju	-jami
PREP	-je	-ax	-je	-ax	-i	-jax

Table 1: Example of Inflectional Paradigms for Russian Nouns. The second row and leftmost column indicate the morphosyntactic features that each slot represents.

These tables are challenging to induce from raw data. Within one language, inflectional classes are not always unique or regular. They may have overlapping patterns and isomorphic morphemes that result in ambiguous forms. For example, an Arapaho verb ending in *-ót* might mean "You alone do something to him/her/it" or it might mean "You alone do something to them" (Cowell and Moss, 2008). Some languages use suppletive forms that have no similarity in shape to their "base" form, or lemma, (e.g. English "go" vs "went"). Field linguists can refer these difficulties to native speakers but computational models are limited to written texts usually sourced from published material.

Monson et al. (2007a) stated two guiding principles for paradigm induction. First, "in any given corpus, a particular lexeme will likely not occur in all possible inflected forms". Even with a large corpus, attempts to recreate a paradigm like Table 1 will result in empty gaps. Linguistics field methods instruction encourages elicitation of inflectional paradigms in focused sessions despite the subfield's emphasis on natural language, precisely because complete paradigms so rarely appear in natural language (Lupke, 2010; Boerger et al., 2016). Some inflected forms will appear much more frequently than others. In fact, it is possible that certain forms may never occur in natural language even though they are grammatically possible (Silfverberg and Hulden, 2018). Even if it is possible to find all the inflected forms of very frequent word, frequent words often follow irregular patterns, as, for example, does the English "be" verb. Paradigms need to be completed by finding overlapping patterns from several incomplete paradigms.

The second guiding principle is that inflected forms of a single lexeme will be similar. This principle does not always hold. Languages abound with exceptions. Word forms may change drastically in a paradigm (e.g. *go* vs. *went*). However, it common enough to serve as a solid working assumption.

In descriptive linguistics, an ideal analysis strives to describe all possible patterns of inflection. Theoretically, this means that every inflectional form of every word needs to be examined, In practice, paradigmatic patterns can be inferred fairly quickly and the linguist can then concentrate on matching lexemes to a paradigm and identifying irregularities. Analysis is assisted by spreadsheets

in computer programs such as Microsoft Excel.

Computational models have successfully induced frequent and regular paradigms with high accuracy even in low-resource settings (Hammarström and Borin, 2011; Durrett and DeNero, 2013; Ahlberg et al., 2014). Most early work on computational morphological paradigm induction applied unsupervised learning to concatenative morphology (Goldsmith, 2001; Chan, 2006; Monson et al., 2007a). Supervised and semi-supervising models have been applied on concatenative and non-concatenative languages (Dreyer and Eisner, 2011; Durrett and DeNero, 2013).

ParaMor (Monson et al., 2007b) is an example of an unsupervised model. It begins by searching the data for strings that resemble inflectional paradigms. It does this by identifying candidate "suffixes" in word-final substrings, like the list in (3b). A candidate suffix is a substring that is found at the end of many different words. The suffixes are refined into sets of partial paradigms ("signatures"), like the one in (3c).

(3) a. **INPUT:** walked, walking, jumps, jumped, ...

b. **Candidate Suffixes:** ed, ing, s, ...

c. **OUTPUT:** ed/ing/s, ...

Inducing inflectional paradigms by unsupervised learning has three flaws. First, a suffix might actually belong to multiple paradigms. The form of suffixes overlap between paradigms. For example, "-s" on an English word may mark noun plurality or it may mark third-person singular subject agreement in the present tense. Second, most candidate paradigms contain "many fewer candidate suffixes than do the true paradigms" (Monson et al., 2007a, p. 903). This is a manifestation of Monson's first principle: induction from natural text leaves gaps because some inflected forms are rare or never used at all. ParaMor solves the first two flaws by leveraging Monson's second principle of consistency of form. It measures the similarity of word forms and uses that similarity score to cluster candidate suffixes into sets that appear on the same or similar roots. The third flaw is that some suffixes will inevitably be identified incorrectly. Morpheme segments will be placed at the wrong place. For example, the English word "ally" could be easily misidentified as

a root “al” plus the adverbializing suffix “-ly”. The third flaw is addressed by filtering paradigms. If a set has only a few suffixes, it is assumed this is due to spurious morpheme identification. A number is chosen and any candidate partial paradigm with suffixes below that threshold are filtered out. So if the cutoff number is 1, then a partial paradigm that only held “-ly” would be disregarded.

Supervised learning of inflectional paradigms focuses on generating or parsing inflected forms. Supervised paradigm induction requires at least partially complete inflectional paradigm tables. Example (4) illustrates possible input and output. The input may include inflected forms, as in (4a), or lemmas (e.g. “walk”) plus the form’s morphosyntactic features. The table is trained on the full data but tested by including missing forms, for which only morphosyntactic features are given. The output would be correct missing forms, as shown in (4b). A parsing model might be trained to output the features in (4a) when given the word forms in (4b).

(4) a. **INPUT:**

walked	PST
_____	PTCP
_____	3SG.PRES
_____	PST
jumping	PTCP
jumps	3SG.PRES

b. **OUTPUT:**

walking
walks

jumped

Malouf (2016) applied supervised deep learning in the form of recurrent neural networks to answer the Paradigm Cell Filling Problem (PCFP) (Ackerman and Malouf, 2009), illustrated in Figure 2, which asks how new speakers infer inflected forms when they have not seen all possible forms? Malouf et al.’s model took as its input the “base” form, or lemma, represented as a one-hot vector, and the morphosyntactic features tags of the inflected form to be generated. For example, if the input was “walk” and the features were tense = PRES, person = 3, number = SG, then the

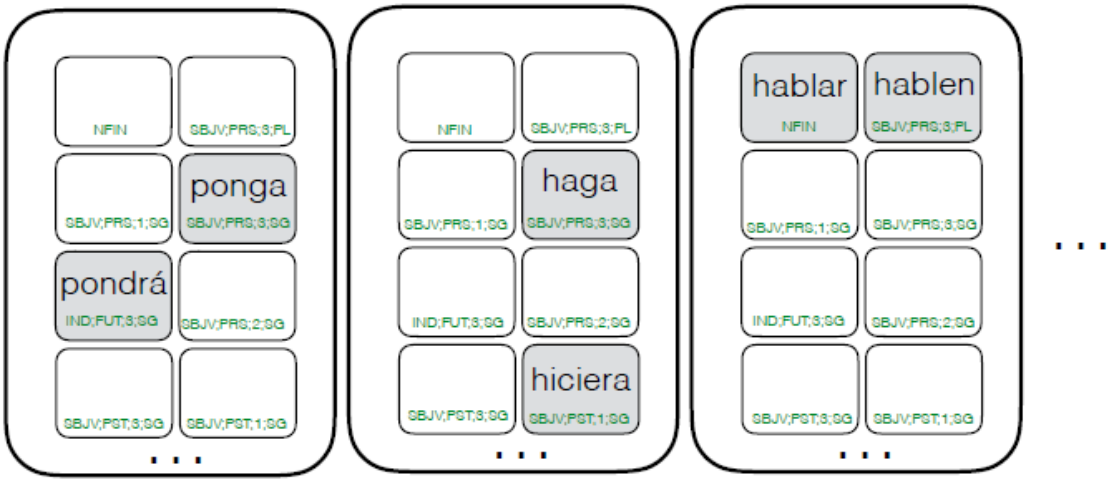


Figure 2: Silfverberg and Hulden (2018): Illustration of the Paradigm Cell Filling Problem with Spanish verb inflectional tables. If speakers are only exposed to such partially filled paradigms, how do they fill the missing cell with the correct form?

correct output would be "walks". Their model was successfully applied to Irish, Maltese, and Khaling [klr], as well as some major languages. One interesting observation is that regularization, which normally keeps the model from overfitting, was found to harm performance because the less frequent inflectional patterns look like noise that regularization tends to guide the model away from, but it actually needs to learn them.

Silfverberg and Hulden (2018) also investigated the PCFP. With the same amount of input their results were very similar to Malouf et al. but their model is more realistic for LDD. They do not use the lemma, which, in morphologically complex languages, is rarely found in naturally-occurring texts. Instead, they trained on inflected forms. Crucially, their input and output were actual strings, not numbers to represent the lemma and one-hot features to represent the inflected form. The current state-of-the-art neural model for PCFP is Liu and Hulden (2020)'s model, which uses the Transformer architecture and leverages principal parts information.

Kann et al. (2016) use a deep learning approach to a similar task called morphological re-inflection. This is essentially the same task illustrated in (4) and the approaches to the PCFP, but without reference to the theoretical question. It is, therefore, not compelled to limit the input.

Kann et al. added encoders that incorporate multiple inflected forms, with the assumption that these multiple forms provide complementary information about a pattern and allows the model to predict a complete paradigm. Such "multi-source" learning performs better than a single data source by working around "holes" in data. It can be applied to fully or only partially annotated data. Kann et al. used complete inflection tables. The model draws inspiration from the theoretical linguistic notion of principal parts (Blevins, 2006; Finkel and Stump, 2007), the idea that there is a minimal subset of forms which allow maximum predictability of related forms. This subset can also be thought of as the smallest number of inflected forms needed to identify the lexeme's inflectional class.

A similar approach has been applied to low-resource settings. Ahlberg et al. (2014) and Ahlberg et al. (2015) used a small number of inflectional tables from Wiktionary. These are are similar to Table 1 but include the whole word. Groups of similarly behaving words were extracted. Several groups were used to identify paradigmatic patterns. As with Kann et al. the immediate goal is to predict the correct inflected forms of unseen words, but the generalized patterns used to make these predictions are basically inflectional paradigms. The patterns are discovered by abstracting the longest common subsequence in a word and clustering the abstract patterns with same or similar patterns. This is illustrated in Figure 3[3]. It seems quite possible that exceptions or irregularities could be accounted for by collapsing similar patterns. The experiment was quite successful on some Indo-European languages (German, Spanish, Catalan, French, Galician, Italian, Portuguese, Russian), as well as Maltese and Finnish.

Attempts at paradigm induction have contributed to computational morphology in low-resource contexts in a singular way: those attempting it have also attempted to estimate the smallest amount of data necessary to perform it with reasonable accuracy. Ahlberg et al. (2014) found that paradigms from the 20 most frequent inflectional classes cover 95% of German word forms. Détrez and Ranta (2012), who use hand-written rules to fill paradigm slots, started with complete inflectional tables and gradually dropped information from them. In the end, they found that

[3]Acknowledgements to Mans Hulden and Ling Liu.

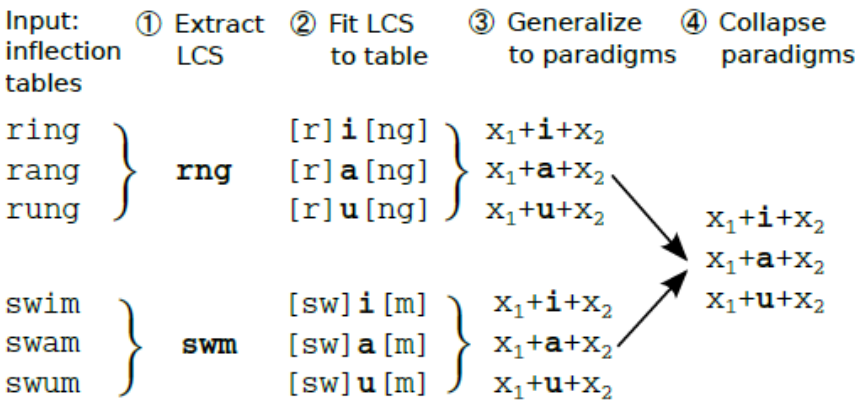


Figure 3: Ahlberg et al. (2015): Abstracting inflectional paradigmatic patterns from inflections and annotated data. The three parts of the longest common subsequence (LCS) *rng* or *swm* are extracted from the inflectional table on the left (step 1) and represented abstractly, in this case as x_1 and x_2. The abstract symbols replace the longest common subsequence in every word form (step 2). In theory, the characters that remain, in this case *i*, *a*, *u*, are the inflectional morphemes. Since the unique part of the each root morphemes is now represented identically, words with the same inflectional patterns will be identical and can be generalized into paradigmatic patterns (steps 3 and 4).

one-two infected forms per paradigm are sufficient to achieve 90% accuracy in English, Swedish, French, and Finnish. Silfverberg and Hulden (2018), using a neural model that learns from a small number of randomly chosen forms per paradigm table, found that two inflected forms gave about 85% accuracy in Finnish, Georgian, Turkish and some Indo-European languages and three forms bumped the results over 90% for all but Georgian nouns and Latvian verbs.

3 Approaches to Computational Morphology

This section explores four major approaches to computational morphology and outlines a history of these approaches. The subsections examine each approach more closely and briefly describe how the approach has been applied to low-resource languages. The four major approaches can be classified as either hand-constructed, rule-based systems (i.e. finite state transducers) or else machine learning systems which "learn" the rules from data. Machine learning can be divided into three strategies based on whether the training data has been annotated completely (supervised), partially (semi-supervised), or not at all (unsupervised).

Until the mid-2000s, the literature on computational morphology was dominated by finite-state machines (Kaplan and Kay, 1981; Koskenniemi, 1983; Beesley and Karttunen, 2003). Gradually, machine learning started to become more popular (Cohen and Smith, 2007; Ruokolainen et al., 2016). Most early machine learning of morphology used unsupervised, probabilistic models (Roark and Sproat, 2007). Supervised approaches came later, due in part to computer memory limitations (Hammarström and Borin, 2011). Semi-supervised learning has grown in popularity more recently, due perhaps to greater number of resources such machine-readable inflections tables becoming available online (Schone and Jurafsky, 2001; Soricut and Och, 2015; Goldsmith et al., 2017).

Currently, supervised end-to-end neural networks, or deep learning, are dominating the field. Although neural networks have achieved state-of-the-art results in many NLP tasks, they did not become popular until recently because they train more slowly than non-neural methods for some tasks (Cotterell and Heigold, 2017). They also require greater computing power. Neural models and methods have greatly improved in recent years. Significantly, an important model for NLP and morphology in particular, the sequence-to-sequence (seq2seq) model, was not developed until a few years ago and the current state-of-the-art Transformer model even more recently. Unfortunately, deep learning models require expertise to customize but linguistics training does not normally include computer programming. As few, if any, good graphic user interfaces exist this presents a practical hurdle to efficient NLP application in LDD.

It is worth noting that comparing and assessing different approaches, and the different models within one approach, is not straightforward. Morphological analysis involves many small steps. If any one step is the basis for evaluation, the assessment changes. A predicted segment can miss the correct morpheme boundary by one letter or by many. Precision and recall can be calculated as a "micro-average" that is measured across morphemes in a word or as a "macro-average" measured across all word forms (Ruokolainen et al., 2013). For macro-averaging, the total count for precision, recall, and F1 measure could be based on types or on tokens. A word type count may obscure whether all types are handled identically. A word token count may make the model seem

less accurate if one very frequent word is incorrectly parsed. A parse may be considered correct when it produces one of all possible ambiguous parses; or it may only be counted correct when it is the right one in the given context (Ruokolainen et al., 2013). Virpioja et al. (2011) examine issues of comparison and assessment over five years of MorphoChallenges and found no solution to these complications, but they concluded that an evaluation based on segmentation is the most simple, robust, and intuitive.

In addition to the difficulties caused by varying methods of comparison between models, computational linguists can use two different ways to evaluate a model. The first is indirect, or extrinsic, evaluation that tries to measure the effect a model has when plugged into a complete NLP pipeline. This is complicated and time-consuming and so rarely done. Direct, or intrinsic, evaluation measures the performance of the system by itself. It can be performed either automatically, measured against a gold standard of annotated data, or manually by domain experts. Virpioja et al. (2011, 53) dismiss manual direct evaluation because human decisions are "subjective" and "the amount of work involved restricts its usage." This disregards the fact that any gold standard data used in automatic evaluation has to be assembled with the same amount of human subjectivity and hard work. Such attitudes reveal the difficulty of discussing computational approaches to linguistic tasks. NLP literature tends to take a narrow focus on how the model performs competitively against other models, rather than how it performs across other corpora and languages.

Computational morphological models and methods have been tested on a growing number of languages since 2000 (Hammarström and Borin, 2011). The different lists of languages between the 2016 and 2018 CoNLL-SIGMORPHON Shared Tasks indicate how this number has grown (Cotterell et al., 2016a; Cotterell and Heigold, 2017; Cotterell et al., 2018). Still, their longest list of about 100 languages is still less than one percent of the nearly 7,000 remaining languages. Nor has this number grown in a representative way. It is not distributed proportionally by language families or basic morphological types. For example, the 2018 CoNLL-SIGMORPHON task included only one polysynthetic language, and that language, Navajo, has been sometimes classified as agglutinative or fusional.

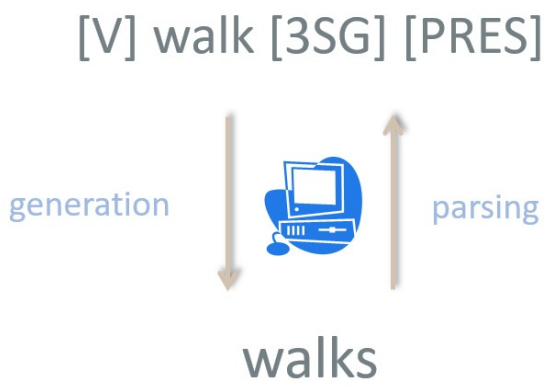


Figure 4: A bidirectional computational morphological model generates an word form from a stem and morphosyntactic tags and parses inflected words forms into stem and tags.

3.1 Finite State Transducers

Finite state transducers (FSTs) are rule-based systems that can successfully modeled linguistic patterns (Koskenniemi, 1983; Beesley and Karttunen, 2003; Hulden, 2009). They were the first successful bidirectional computational model (Goodman, 2013), illustrated in Figure 4. Their ability to represent regular relations between underlying and surface forms, moving left-to-right gave them early importance for modeling Chomsky and Halle (1968)'s rewrite rules and the theory of two-level morphology (Karttunen and Beesley, 2005).

FSTs remain important to computational morphology. As recently as 2017, it was claimed that computational morphological learning is essentially still finite state methods (Goldsmith et al., 2017). For many applications they have been found superior over a feature-based, semi-CRF supervised machine learning model (Cotterell et al., 2015).

The most popular methodology to construct FSTs has two steps. The first step specifies a lexicon and morphotactics in a finite-state lexicon compiler (*lexc*), illustrated in Figure 5. This is where concatenative morphological rules and morphological irregularities are addressed. The second step implements morphophonogical rewrite rules. These rules apply changes in specified contexts and allow the FST to move beyond simple concatenation of morpheme strings to the generation of well-formed inflected forms. For example, a rewrite rule would specify that the final letter in the Arapaho stem *noohow* 'see' in the *lexc* transforms into a *b* if a vowel-initial morpheme follows it.

```
LEXICON Stem
...
@U.StemType.TA@@U.StemInitial.Yes@noohow@U.StemFinal.Yes@:
@U.StemType.TA@@U.StemInitial.Yes@^noohow^@U.StemFinal.Yes@          OrderInflection;
...
```

Figure 5: A snippet of an Arapaho *lexc* file for one stem morpheme *noohow* 'see'. The @ surround flag diacritics for long-distance concordance patterns. This notational addition to basic FSTs encode what linguists might "feature unification". It signals the rewrite rules how to handle transformations unique to a stem's inflectional class or other exceptional morphophonological behavior.

FST "grammars" take significant effort to develop, maintain, and update (Durrett and DeNero, 2013; Moeller et al., 2018). The grammar must be manually-constructed, so developing FSTs requires a thorough knowledge of both the language and finite state technology. Although in at least one case a FST morphological analyzer was constructed with only knowledge of labeled inflection tables, it was limited to the analysis of some inflectional morphology (Forsberg and Hulden, 2016). The bulk of a FST can be developed relatively quickly with basic documentary and descriptive resources (e.g. the Boasian Triad), but its reliance on language-specific lexicons and rules means that an effective FST grammar needs thorough descriptive resources.

With enough linguistic expertise and time for development, FSTs are capable of correctly analyzing any well-formed word, but FSTs cannot be made to generalize to unseen forms. Sometimes it is practical for downstream tasks to have just the one "best guess" that the FST gives. This best guess can be improved by forcing the FST to rank analyses with added weights (Roark and Sproat, 2007). For example, adding weights to rare morphemes if needed. These weights can be added by hand from linguistic analyses or they can be learned by an unsupervised algorithm such as Expectation Maximation (EM). However, weighted FST are not commonly used for morphology.

If both descriptive resources and a language expert trained to construct the *lexc* and rewrite rules are available, an FST can be a good solution for simple morphological modeling. FSTs work well with languages that have strong constraints on the inflections of each lexical category, or part of speech (POS), such as Arapaho which has unique polysynthetic verb forms. However, FSTs do not help resolve ambiguity between wordforms because they output all possible outputs instead of

ranking probabilistically given the surrounding context. FSTs cannot be used for active learning very easily because they do not learn from annotated examples. These issues can be resolved with machine learning.

3.2 Unsupervised Machine Learning

Unsupervised (computational) learning of morphology (ULM) began as an attempt to prove American structuralism and Bloomfield's "inductive generalizations," as well as a possible language acquisition model (Hammarström and Borin, 2011). ULM models attempt to induce morphological patterns from raw, unannotated texts. They take as input natural language data and, with as little supervision (i.e. parameters, thresholds, human intervention, model selection, etc.) as possible, outputs a "description" of the language's morphological structure, according to Hammarström and Borin (2011). However, their use of the word "description" is a bit misleading. What ULM produces is only a first step to linguistic description. Settles (2010, pp. 33-34) depicts ULM more accurately as exploiting the "latent structure [in the data] alone to find meaningful patterns." It outputs the data organized "in a meaningful way," usually via a clustering algorithm that finds natural groupings within the data. For example, one natural grouping would be words that allow the endings "-ed/ing/s".

According to Monson et al. (2008), unsupervised morphological models have followed three main stages of development. In the first stage, early unsupervised algorithms drew inspiration from Z. Harris (Harris, 1955; Harris, 1967). Harris extended a descriptive methodology pioneered by Bloomfield that focuses on discovering what elements in a language can co-occur or not. Harris observed that phonemes do not co-occur unpredictably. The probability of a phoneme appearing in a word is dictate by the previous phoneme, the third is dictated by the first two, and so on. Harris (1955) was the first to use character predictability to identify morpheme boundaries. This is illustrated in Figure 6. In the second stage, models inspired by Harris implemented the Minimum Description Length principle (MDL). At the third stage of ULM development, models began to leverage patterns akin to inflectional paradigms in order to find inflectional relationships between words and identify affixes.

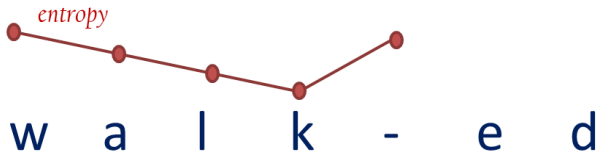


Figure 6: Harris' character predictability/entropy predicts a morpheme boundary where the uncertainty (entropy) about the next letter increases. In "walked", the possible letters that can follow the "k" are more than those that could follow the "l", so we predict a morpheme boundary.

MDL remains a leading method for unsupervised learning of morphology (Hammarström and Borin, 2011). It extends the Kolgomorov complexity that asks, in essence, "What's the shortest computer program (in terms of memory) that generates some text?" MDL, illustrated in Figure 7,[4] simplifies this somewhat abstract question into a weaker, but computable, model. MDL-based approaches attempt to model morphological structures by minimizing the combined memory of the input data and the model. A model would encode hypothesized morpheme strings with binary codes. This is essentially a lexicon or "codebook" of morpheme candidates. Going through the data, the model replaces the morpheme strings with their codes. Then the memory cost of the "codebook" and the encoded texts are calculated. The best morphological model is the one that where the "codebook" plus the encoded texts have the smallest possible memory cost. Measuring the cost of the model reduces overfitting. Measuring the cost of the data, where the most frequent morphemes have the shortest codes, encourages frequent strings to be identified as morphemes. Two downsides to MDL are that it provides no hint where the model should start looking for morpheme boundaries (what hypothesis it should start with) and it cannot it provide a coherent analysis across a whole language (Goldsmith et al., 2017).

Leading MDL-based models are Linguistica (Goldsmith, 2000; Goldsmith, 2001) and the Morfessor family of algorithms (Creutz and Lagus, 2005; Creutz and Lagus, 2007b). Linguistica is a foundational work in ULM that attempts to discover "signatures", shown in Figure 8 and (3), that are sets of affixes somewhat akin to inflectional paradigms. Even though it is unsupervised, Linguistica needs to set an affix length limit and this requires at least enough knowledge of the

[4]Generated at https://asr.aalto.fi/morfessordemo/

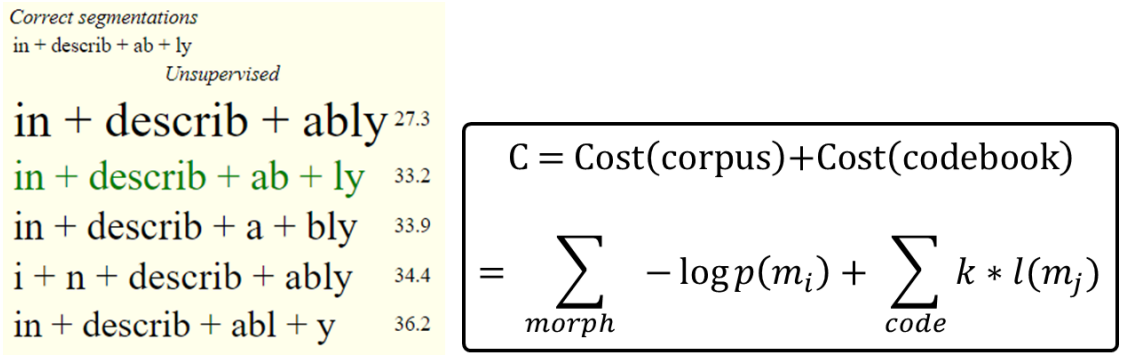
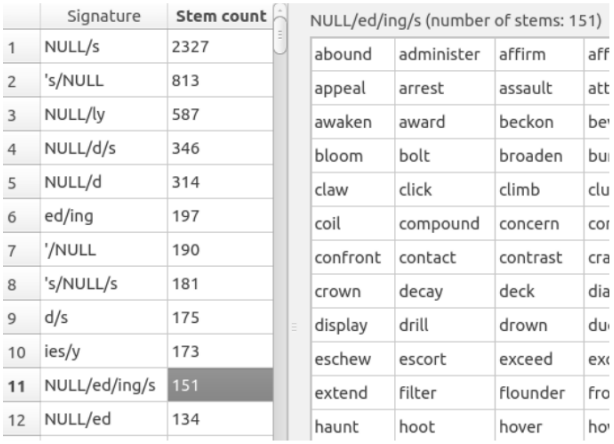


Figure 7: Minimum Description Length. The model makes random splits (marked by + on left) in a corpus, resulting in strings of candidate morphemes, or "morphs". A "codebook" of binary codes that represent each morph is constructed so that more frequent "morphemes" are represented with shorter codes, reducing the average cost per morpheme. Then each occurrence of a morph in the corpus is replaced with its code. Finally, the total cost *C* of the model (codebook) and the data (encoded corpus) is summed. On the left, this cost *C* is shown next to the word for a model with various segmentations of "indescribably". The random splits that result in the lowest total cost is considered the correct morphological model. The "wrong" lowest score here demonstrates typical segmentation errors in unsupervised learning of morphology. The cost of the encoded corpus is calculated by summing the negative log likelihood of each morph's maximum likelihood $p(m_i)$ (i.e. the count of a morph's m_i occurrences divided by the total number of morphs in the corpus). The codebook cost is calculated by multiplying the number of the bits *k* needed to code a character by the character length of each morph $l(m_j)$.

language's morphology to make a hypothesis about its longest affixes. The latest version, Linguistica 5 (Lee and Goldsmith, 2016), attempts to make ULM more accessible by including a graphical user interface and an open-source, modular Python library. The library is meant to extend the model beyond morphology, taking syntactic context into account (Nicolai and Kondrak, 2017). Morfessor (Creutz and Lagus, 2005; Creutz and Lagus, 2007a; Creutz and Lagus, 2007b) attempts to identify the most likely "morphs" (candidate morpheme strings) and the most likely morpheme boundaries. Like many unsupervised models, it builds a morph lexicon. Relying on a lexicon of morphs reduces a model's cross-linguistic adaptability because the lexicon is specific to each language. The Morfessor model is built on two parameters that require some hypothesis about the language's morphology: "(i) our prior belief of the most common morph length, and (ii) our prior belief of the proportion of morph types that occur only once in the corpus" (Creutz, 2003, p. 281). Compared to Linguistica which is less eager to split words, earlier versions of Morfessor



	Signature	Stem count
1	NULL/s	2327
2	's/NULL	813
3	NULL/ly	587
4	NULL/d/s	346
5	NULL/d	314
6	ed/ing	197
7	'/NULL	190
8	's/NULL/s	181
9	d/s	175
10	ies/y	173
11	NULL/ed/ing/s	151
12	NULL/ed	134

NULL/ed/ing/s (number of stems: 151)

abound	administer	affirm	aff
appeal	arrest	assault	att
awaken	award	beckon	be
bloom	bolt	broaden	bu
claw	click	climb	clu
coil	compound	concern	cor
confront	contact	contrast	cra
crown	decay	deck	dia
display	drill	drown	du
eschew	escort	exceed	ex
extend	filter	flounder	fro
haunt	hoot	hover	ho

Figure 8: Screenshot of Linguistica 5. The column on the left displays hypothesized "affixes" that were found in complementary distribution on "stems", one group of which is displayed on the right.

used Recursive MDL which tended make too many segments, but later versions of Morfessor stand somewhere in between early Morfessor and Linguistica in terms of segmentation accuracy (Creutz, 2003).

ParaMor is another unsupervised algorithm (Monson et al., 2004; Monson et al., 2007b; Monson et al., 2007a; Monson et al., 2008). Like Linguistica, it exploits patterns in the data similar to inflectional paradigms, but unlike Linguistica, it allows multiple segmentations per word. Paramor treats paradigm induction and segmentation separately. It does not address morphophonology as well as Linguistica and requires more data though not as much memory. Monson et al. (2008) claim that Paramor's unsupervised induction of morphology could facilitate quick development of morphological learners for low-resource languages, but they refrain from explaining how the system, which requires a significant amount of data, could do this.

Most ULM approaches are biased towards a certain affixation pattern or morphological type. Linguistica and ParaMor are tuned to suffixing morphology. Morfessor is better at prefixing+suffixing languages than the other two, but Linguistica does better with suffixing-only languages. Both Linguistic and Morfessor do not handle non-concatenative morphology, such as the templatic morphology of Semitic languages. All three are most suitable to agglutinative languages but also assume a small number of morphemes per word, often only one prefix/suffix. This does

not bode well for morphologically complex languages.

In theory, unsupervised models are the most exciting and attractive for LDD because they do not require costly data annotation. In fact, an early vision of ULM was that it could support language technology for minority language communities. This idealistic aim has not materialized because successful ULM demands such a large amount of unannotated words (Ruokolainen et al., 2016). Languages that lack basic morphological description generally lack the amount of raw data that accurate ULM requires. "Sufficient" data for unsupervised learning is on the order of a hundreds of thousands or a million words (Rocio et al., 2007). A small corpus is defined as 1,000-100,000 words (Kirschenbaum et al., 2012). Since LRL includes well-documented languages, sometimes obtaining sufficient data may be simply a matter of digitizing existing literature, but for most, a limited written history means that texts must be first recorded orally and then transcribed before this a small corpus is usable. Although language documentation generates minimally annotated data (Himmelmann, 1998; Lehmann, 1999), LDD projects rarely produce even a 100,000 words and the annotated portion tends to be much smaller. Nevertheless, the claim is still being made that ULM methods, "provide an inexpensive means of acquiring a type of morphological analysis for low-resource languages" (Ruokolainen et al., 2016, p.92).

Another drawback to ULM is that unsupervised methods are less accurate than other machine learning approaches. Soricut and Och (2015) were able to overcome this drawback, at least for some Indo-European languages by using unsupervised word embeddings (word meanings represented in vector space). However, even if word embedding increase accuracy, it is an open question whether they can address the data requirements because successful word embeddings are also built with large amounts of texts.

Goldsmith et al. (2017) raise other problematic issues. For example, unsupervised approaches have to start with some assumptions about a language's morphology. Should we start with random parameters or should we use knowledge about the target language? The latter is the most efficient, but using prior linguistic knowledge, as Goldsmith, Lee, and Xanthos and Palmer et al. (2010) insinuate, means the approach is not truly unsupervised. It also makes ULM even less attractive

for under-described languages. With such languages, we can presumably hypothesize from related or geographically adjacent languages, but this opens another question that is not clearly addressed in the literature: how well do current ULM methods transfer across languages, and can we even measure this reliably? This cannot be addressed until ULM models have been tested against a wider range of languages. Currently, English is the *de facto* language in ULM (Palmer et al., 2010). The tension between expanding computational models to a wider range of languages and the prohibitive cost of data to test and evaluate the models will continue to influence the development of computational linguistics unless better data production methods are devised.

Although unsupervised machine learning is most suitable to resource-rich languages, ULM has also been applied in low-resource contexts. Palmer et al. (2010) incorporated unsupervised morphological segmentation as a first step towards semi-automated interlinearization in Uspanteko [usp], a Mayan language of Guatemala with some 4,000 speakers (**?**). Words were assumed to be morphologically related if they were orthographically similar. Affix candidates were generated by assuming stems are longer than affixes and then filtering for statistically significant co-occurrences. Moon et al. (2009) also applied ULM to Uspanteko with a method that assumed suffixation only and, most notably, incorporated awareness of document boundaries when generating and clustering candidate morphemes. This technique assumes spelling is likely to be consistent within a document but vary across documents. Considering that minority languages may not have a standardized orthography and transcribers may have little formal education in the language, this is a simple yet very practical twist on the "one sense per discourse" effect. This is an observation used in computational semantics that notes if a polysemous word such as "bank" appears more than once in a document all occurrences are highly likely to express the same sense (Gale et al., 1992). Document boundary awareness assumes one inflectional paradigm per stem per document; that is, a morpheme that may belong to two lexical categories is highly likely to appear as only one within a single document. This technique reduces noise and improves results over Morfessor and Linguistica (for English). Remarkably, the performance degrades when the corpus size is increased, perhaps because of increased noise from spurious candidate morphs. Kirschenbaum et al. (2012)

applied ULM to a corpus of Kilivila [kij], an Austronesian language of Papua New Guinea with 20,000 speakers (Eberhard et al., 2020). Inspired by Schone and Jurafsky (2001) and Baroni et al. (2002), they identified morphologically related words by orthographic similarity and a context co-occurence vector. They claim that the method prefers languages with infixes but it is unclear how they determined this with only one language.

3.3 Supervised Machine Learning

Supervised learning of morphology means that models are trained on gold standard annotated data rather than unannotated texts. It requires significantly less data than unsupervised learning. The need for annotated data makes supervised machine learning qualitatively different than unsupervised learning. Unsupervised learning clusters underlying patterns or features which may facilitate human analysis and description of the data. Supervised learning is trained on data that has already been analyzed and labeled, as in (5a). It learns to generalizes the annotation, so that it can predict with high accuracy the labels, as in (6b), of unseen instances like (6a).

(5) a. **Training Data:** walked *walk.PERAMBULATE* ed.PAST

b. **Features:** tense=PAST, POS=V, previousword=X, nextword=Y, ...

(6) a. **INPUT:** jumped

b. **OUTPUT:** jump.HOP ed.PAST

The best performing non-neural supervised model for sequence prediction tasks such as morphological analysis is conditional random fields (CRF) (Lafferty et al., 2001; Müller et al., 2013; Ruokolainen et al., 2016). A CRF is a sequence classifier that considers the whole sequence when making a prediction for each symbol in the sequence. In morphology, CRFs can work as boundary detection (segmentation) and labeler. According to Ruokolainen et al. (2016), discriminative learning methods such as CRF are better than generative models because they optimize the accuracy of segmentation boundaries and generalize better to unseen forms, assuming they have sufficient training data. A discriminative model's strength lies in its ability to define features beyond the

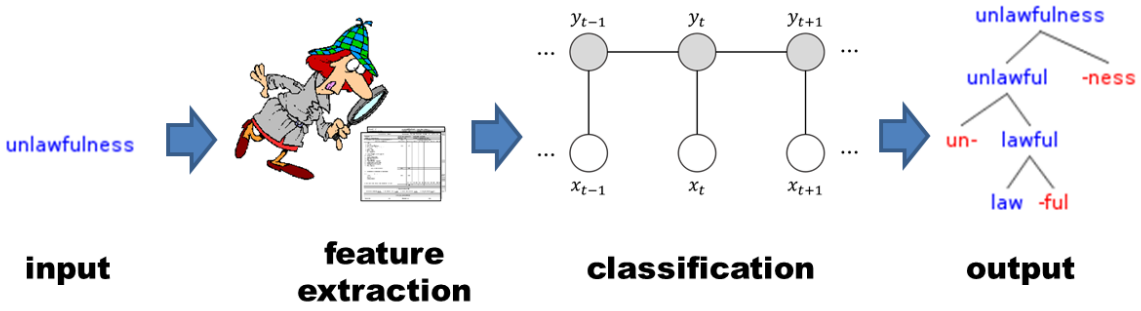


Figure 9: Classical machine learning. In feature-based models a human expert must analyze and define optimal features from the data. A CRF usually uses gradient descent to assign weights to those features during training. A discriminative model such as Conditional Random Fields, can assign arbitrary weights.

previous label and allow arbitrary weights. CRFs apply logistic regression which allows it to make generalized predictions from arbitrary and possibly dependent features (Ruokolainen et al., 2013).

A CRF can look at a number of features, such as those in (5b) as it makes a prediction. A feature function input for a (linear-chain) CRF in morphological labeling might be 1) a whole word segmented into morphemes, 2) the position of the current morpheme in the word, and 2) the label of the previous morpheme. Each feature is weighted during training and, with these weights, labeling predictions are scored over the whole sequence, then transformed into a probability. For example, in a segmentation task where the previous word is "am/is/are/was/were", if the target word ends in "ing" and a weighted feature is the previous word the CRF might give a high weight to "-ing" as a separate morpheme. In labeling, the CRF might decide a word-final "s" marks a plural noun and not the 3rd person singular simple present tense by highly weighting the POS tag of the stem. However, the quality of the results relies heavily on the choice of features. This could be a drawback for under-described languages, because if little linguistic description is available then how does one know which linguistic features are optimal? Fortunately, CRFs have been shown to perform reasonably well using primarily language-independent features (i.e. surrounding substrings) (Ruokolainen et al., 2016; Moeller and Hulden, 2018), including a model trained on approximately 3,000 words of Lezgi [lez], an agglutinative Nakh-Daghestanian language with about 600,000 speakers (Eberhard et al., 2020). However, it is not clear if the performances were

more dependent on language-specific or task-specific features.

Supervised (and semi-supervised, see section 3.5) learning requires less data and almost always achieves better results than unsupervised learning (Ruokolainen et al., 2013) and, therefore, seems more promising for LDD. However, it may be necessary to augment textual data with other descriptive resources such as grammars and dictionaries since the size of annotated LDD corpora are still "inadequate for supervised learning" (Duong, 2017, p. 18). This is not an impractical expectation for LDD since descriptive linguists traditionally result in the Boasian triad, described in Figure 10.

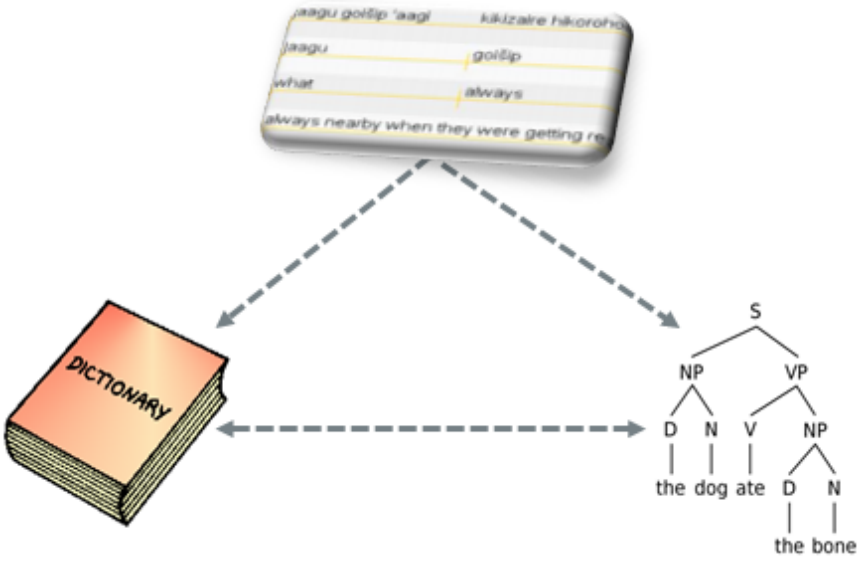


Figure 10: The Boasian Triad. Traditionally, linguists working in new languages concentrate on producing a corpus of interlinearized texts, a grammar, and a dictionary. The interlinearization is a pre-step and feeds directly into the other two steps.

3.4 Supervised Neural Networks (Deep Learning)

Neural networks (NN), or deep learning, is a family of machine learning techniques that builds layers of perceptrons that can learn complex patterns. Vector representations of the data are received by an input layer, fed into and transform in "hidden" layers then arrive at the output, or activation, layer. Each unit in each layer is connected to each unit in the adjacent layers. The connections are represented by learnable weights; the higher the weight the more influence a unit has on the

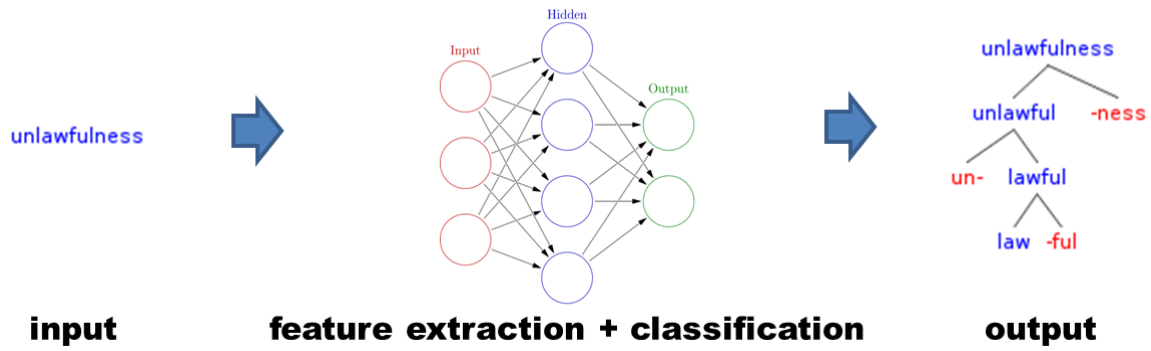


Figure 11: In deep learning, hidden layers create intermediate representation of the data. These essentially serve the function of feature engineering in non-neural machine learning.

result. Since deep learning is supervised[5] the weights are adjusted via stochastic gradient descent with backpropagation using "feedback" from annotated data. Many flavors of deep learning exist and different models work best for different tasks. Recurrent neural networks (RNN) (Elman, 1991) with long short-term memory gates (LSTM) (Hochreiter and Schmidhuber, 1997) are currently considered state-of-the-art for most sequence-based NLP tasks. RNNs may have one layer and recurrently add its output to next input and feed that into that layer. Many may have multiple layers and the recurrence occurs only for each time step in the network. This works as a feedback loop. RNNs can input and output sequences of values. A probabilistic prediction of the next item in a sequence is conditioned on the entire sequence of transformations and previous predictions in each time step. Unfortunately, the typical deep learning problem of vanishing gradients, or decay of information through time, is much worse for RNN. That is why the LSTM memory gate is so important. The LSTM decides how much of the previous output should feed into the next recurrence and how much should be "forgotten". Dropping forgotten information from the calculation allows RNNs to propagate the gradients much more efficiently back through the entire sequence of time steps.

In morphology, deep learning methods can be used to classify, generate sequences, or to performs sequence-to-sequence mapping. Classifiers classify data points into categories or classes. A data point might be a morpheme and the classes morphosyntactic tags. The hidden layers feed

[5]Deep learning morphological segmentation has been performed on unsupervised texts with some success (Wang et al., 2016)

into a final logistic function layer (i.e. softmax) that outputs a prediction of each possible class as a probability between 0 and 1. Morphological generation produces fully inflected forms from morphosyntactic or contextual information (Cotterell et al., 2017) or completes morphological paradigms (Malouf, 2016).

Currently, the strongest generator models use an encoder-decoder, also known as sequence-to-sequence (seq2seq). The Transformer (Vaswani et al., 2017) architecture seems to be reaching state-of-the-art in most tasks. Before that, the architecture of a RNN (Sutskever et al., 2014; Kann et al., 2016). Encoder-decoders map one sequence to another of a different length by encoding an input sequence of symbols (e.g. the letters of a word) and decoding it as another sequence of symbols (e.g. morphosyntactic tags). Encoder-decoders have been state-of-the-art for morphological tagging (Heigold et al., 2017), morphological segmentation (including low-resource settings) (Kann et al., 2018), and morphological paradigm completion (Cotterell et al., 2018).

Until recently, neural networks were considered to have the same drawback as unsupervised learning: large amounts of training data for good results (Cotterell et al., 2018). While it is true, for example, that top performing neural networks generate inflected forms with less than 60% average accuracy when given only 100 training examples, results can increase significantly (up to 96%) when trained on 1,000 and 10,000 examples (Cotterell et al., 2016b; Cotterell et al., 2017; Cotterell et al., 2018). Improved results cannot always be accomplished by the tuning tricks common with normal data sizes. For example, adding hidden layers can actually reduce accuracy with smaller amounts of data (Cotterell et al., 2017). Low data settings require unique techniques that include adjusting the model, training an intermediate step, and artificially augmenting the data. As an example of adjusting the model, Sudhakar and Singh (2017) found that using a gated recurrent unit (GRU) to control access to previous states in the RNN, rather than the NLP state-of-the-art LSTM, worked better with up to 1,000 training examples. The most significant recent change in model has been the Transformer (Vaswani et al., 2017) which has become the state-of-the-art model for character-level NLP tasks (Wu et al., 2020).

Successful intermediate steps in morphology include training the model to make edits to a string

instead of decoding a new sequence. For example, the model learns an INSERT operation generates "runs" from "run" (these models use an alignment algorithm to reduce the number of edit operations between related forms) (Makarov et al., 2017; Makarov and Clematide, 2018). Other successful intermediate steps include the whole sentence where an inflected form is or should used or train a model to first produce abstract underlying forms (e.g. "impossible" → "in-possible" → "NEG-possible") the final inflected or parsed form (Liu et al., 2018; Moeller et al., 2019). In recent CoNLL-SIGMORPHON submissions, participants found that these, and other new techniques considerably outperformed the state-of-the-art encoder-decoder baseline (Bergmanis et al., 2017). These successes suggests that developing unique techniques for morphological analysis of low-resource languages is a promising area for exploration. A few of these techniques are discussed further in section 4.3.

3.5 Semi-Supervised Machine Learning

Unsupervised learning may require less costly annotated data than supervised learning and supervised training may beat unsupervised training in accuracy, but combining the two could be the best of both worlds when the available annotated data is limited. Semi-supervised, or minimally supervised, learning combines training on some annotated data with a larger set of unannotated data (Kohonen et al., 2010; Poon et al., 2009). Like unsupervised learning, semi-supervised learning exploits the underlying structure of raw texts, but with the goal of improving the machine's predictions, rather than discovering new ones (Settles, 2010).

Semi-supervised learning has been widely researched for computational morphology and a wide variety of techniques have been employed (Ruokolainen et al., 2016). Some ULM algorithms, such as Morfessor Baseline, have proven adaptable to semi-supervised learning (Kohonen et al., 2010). Dreyer and Eisner (2011) used a "mostly unsupervised" method that employs a generative probabilistic model and 10 million unannotated words. It does POS-tagging, segments and tags morphemes, and identifies candidate inflectional paradigms. Ahlberg et al. (2014) predicted general paradigmatic patterns using longest common substrings and some unannotated data (LCS) (see section 2.2 for more details). The latter two models presuppose a few completed inflectional

paradigms which are provided by native speakers either directly, or as computational linguists tend to prefer, through the indirect means of an expert who pre-processes the data or through crowd-sourced resources such as Wiktionary. Liu and Hulden (2020) used semi-supervised learning to improve paradigm induction.

Unfortunately, although semi-supervised learning was early praised for its greater effectiveness and practicability, real applications were rare well into the late 2000's (Druck et al., 2007). One reason may be that, like unsupervised learning which requires some initial hypothesis, many semi-supervised models make strong assumptions about the data. These assumptions may hold true in pre-processed datasets but "tend to be violated in real-world data" (Druck et al., 2007, p. 1). In addition, unannotated and annotated data cannot be simply combined together. Annotated data has to be weighted so that the larger, unannotated part does not overwhelm it. Overall, results in semi-supervised learning "strongly suggest that it is crucial to use the few available annotated training instances as efficiently as possible before…incorporating large amounts of unannotated data" (Ruokolainen et al., 2013, p. 35).

Semi-supervised learning is promising for LDD because even though small amounts of annotated data are "easy to get by manual annotation" (Virpioja et al., 2011, p. 49), typical documentation corpora are only partially annotated. However, the results suggest that all available descriptive data for LRL should be exploited as much as possible before relying on unsupervised data. This heavy dependence on annotated data still poses a challenge (Andrews et al., 2017).

4 Exploiting Resources

LDD is unfortunately plagued with spotty and noisy data. Even though supervised and semi-supervised methods requires less data than unsupervised or neural network models, they still need significantly more data than is typically available for low-resource languages. So, building a successful NLP system for LRL necessitates knowing what resources are available for the target LRL (Duong, 2017) and finding techniques that make best use of these resources (Palmer, 2009).

This section explores techniques that have shown to improve results in machine learning ap-

proaches to morphology. These techniques that exploit resources other than textual data, including artificially generated "textual" data, descriptive linguistic knowledge, and resources in other, higher resource languages. Most of these techniques have been developed specifically for low-resource contexts.

4.1 Non-Textual Resources

In practice, language documentation projects are rarely undertaken (and more rarely funded) except to support descriptive analysis and publication (Thieberger, 2012; Austin, 2014; Vallejos, 2014; Thieberger et al., 2016). Even though annotated textual data is scarce for most languages, many have published descriptions of some sort. Those publications contain linguistic resources. Describing a language's morphological structure is relatively easy (Roark and Sproat, 2007) and linguists tend to tackle it in the first stages. Minimally, a few inflectional paradigms can be easily elicited in a language documentation project. An extensive description of the language's morphology may be published as organized, structured data. This includes inflectional tables and word lists.

Crowdsourced websites such as Wiktionary sometimes have inflectional tables for LRL. Cotterell et al. (2015) successfully exploit spell checkers and Wiktionary to train a semi-supervised model. Ahlberg et al. (2014) use publically available inflection tables with a feature-based Support Vector Machine (SVM) to classify unseen lexemes into inflectional classes. Durrett and DeNero (2013) also exploit Wiktionary for inflectional tables as the supervised part of a semi-supervised model. They claim that this allows their model to "extend to other languages [with inflectional tables in Wiktionary] without change" but it is to be expected some of the 150 or so languages on Wiktionary have very few tables.

If data is thoroughly annotated, computational morphology can improve results from labeling that are not strictly morphological in nature. For example, Müller et al. (2013) reduced CRF training time by performing a coarse POS-tagging then fine-grained morpheme tagging. The POS-tagging leaves the algorithm with fewer possible tags to process for each sample.

Exploiting non-textual resources does not always improve results. Semi-supervised learning

with a discriminative sequence learner like a CRF performs worse when non-textual resources are integrated because the external resources receive undue weight. Andrews et al. (2017) claim that models, such as a Maximum Entropy Hidden Markov Model, that generate a lexicon are less likely to degrade in performance because the lexicon essentially acts as an external resource and does not compete with the annotated corpus.

4.2 Transfer and Joint Learning

Transfer and joint learning share a common big idea. They attempt to improve computational models by "borrowing". Transfer learning borrows the success of other models. Joint learning borrows information from other areas. Generally speaking, transfer learning assumes that what succeeds in one task will work well on another and it attempts to transfer that success. Transfer learning can be applied in semi-supervised and unsupervised learning (Duong, 2017).

When the task involves generalizing to another language, it often referred to as cross-lingual learning instead of transfer learning. Examples of cross-lingual learning are applying a successful morphological model to a new language or re-purposing annotated tags by aligning words across languages (Duong, 2017). Generalizing across languages works better when a model is trained on more than one language. Kann et al. (2018) found that single segmentation model trained simultaneously on four related polysynthetic languages performed as well as, and sometimes better than, single language models. However, more recently the SIGMORPHON 2020 task 0 baseline with a Transformer architecture found across 90 languages that transfer learning across related languages did not improve performance, and sometimes was a disadvantage.

Baumann and Pierrehumbert (2014) applied transfer learning principles to computational morphology by using an English word list to identify affixes in unlabeled data of Tagalog [tgl] and Zulu [zul], low-resource languages (but clearly not endangered – both have over 20 million speakers) spoken in the Philippines and South Africa respectively. The strategy was motivated by the realization that many low-resource languages are spoken in multi-lingual contexts and have borrowed vocabulary from more economically powerful languages. A word list in that language identifies some root morphemes. Splitting substrings from these roots provides a list of prefixes and suffixes.

With some knowledge of the language's morphological patterns, it was even possible to identify infixes in Tagalog. However, the strategy does not do well when a word has multiple affixes.

Transfer learning is promising for low-resource languages but it has its limitations. When RNNs performed cross-lingual morphological tagging on 18 languages from four language families, the conclusion was informative but perhaps unsurprising to those familiar with linguistic typology—the closer the languages' relationship, the more accurate were the results (Cotterell and Heigold, 2017). The precise correlation between linguistic genetics and results in transfer learning is largely unexplored (Buys and Botha, 2016; Cotterell and Heigold, 2017), although interest is growing and new experiments are forthcoming (McCarthy et al., 2019). It is still not clear, for example, whether morphological structure or phonological (orthographic) similarity is a stronger factor. It *is* clear that the amount of morphologically marked information common across languages influences results, so if a word "in the source language does not overtly mark a grammatical category [that is marked] in the target language, it is nigh impossible to expect a successful transfer" (Cotterell and Heigold, 2017, p.749). Transfer learning shows sharply the difficulties in paradigm induction. Paradigms do not generalize across languages. Each language has its own unique "layout" for classes within each lexical category and these can differ considerably even in closely-related languages.

Since transfer learning can potentially overcome deep learning's data hunger, discovering how to improve deep learning results in when transferring from high resource to low-resource settings would be "a boon for low-resource computational linguistics" (Cotterell and Heigold, 2017, p. 752). Duong (2017) points out that automatic projection of annotation (specifically, POS-tagging, noun phrase chunking, and dependency parsing) between high and low-resource languages admits error at multiple points. The results have to be corrected or adjusted. Any projection-based method requires some seed data to perform well (Buys and Botha, 2016), so one question ripe for exploration is how much data does the low-resource target language need to balance a high resource source language's greater data? Cotterell and Heigold (2017) claim only a "small amount of annotation" is required; but they also claim the necessary annotation should not take too long.

Conspicuously, they refrain from defining “small amount” and “not too long”.

Joint learning trains simultaneously on different types of linguistic information. For example, since segmentation and tagging are generally separate tasks in computational morphology, training a semi-CRF to do both tasks would be considered joint learning (Cotterell et al., 2015). It is based on the intuition that one type of linguistic knowledge (e.g. syntax) can improve results in another domain (e.g. morphology), and vice versa (Goldsmith et al., 2017). One of the earliest joint learning attempts was Schone and Jurafsky (2001) who incorporated semantic, orthographic, and syntactic information into unsupervised learning of morphology. The semantic information came from Latent Semantic Analysis which represents meanings as patterns of words that appear together in text, for example “morphology”, “morpheme”, “inflection”’. Semantic information helped avoid the typical unsupervised segmentation error (e.g *all-y* instead of *ally*). It was supplemented by the probability of two affixes alternating on one “stem”, which was calculated using orthographic information. Syntactic information was derived from the frequency of words that occur around pairs of possibly related inflected forms. The output was “conflation sets” which are paradigm-like groups (e.g. “abuse”, “abused”, “abuses”, “abusing”).

Since morphology carries a great deal, and in some languages most, syntactic information, syntax is important information for morphological analysis. Joint learning of syntax and morphology can be as simple as incorporating POS tags as features or extending POS tagging methods to morphological tagging (Buys and Botha, 2016; Cotterell and Heigold, 2017). Syntax can also be incorporated into supervised learning by clustering words with similar final substrings which serve as a proxy for grammatical agreement and are assumed to indicate lexical categories (Lee et al., 2011). Successful joint learning of semantics and morphology has used the semantic information in word embeddings (Soricut and Och, 2015). Joint learning in low-resource settings has been successfully applied using information commonly produced by LDD projects (Palmer, 2009; Moeller and Hulden, 2018), particularly POS tags.

4.3 Data Augmentation

Another "resource" that can be exploited is artificially generated data which augments naturally occurring textual data. Bergmanis et al. (2017, p. 38) found that the "main benefit of the various data augmentation methods is providing a strong bias towards ... regularizing the model, with a slight additional benefit obtained by learning the typical character sequences in the language." Kann et al. (2018) found that with data augmentation a neural model can outperform Morfessor and match CRF accuracy. The success of these methods suggests that data augmentation may be a fruitful area of experimentation for applying neural models in low-resource settings.

The most successful data augmentation strategies in low-resource settings bias an encoder-decoder model to copy strings (Bergmanis et al., 2017; Kann et al., 2018; Makarov et al., 2017; Makarov and Clematide, 2018). For example, Bergmanis et al. (2017) and Kann et al. (2018) auto-encode (train a model to output a string identical to the input) words from the training corpus. Their performance varied between languages, perhaps due to how well a language's morphological features were represented in the small training datasets, but Bergmanis et al. (2017) found this method performed better than transfer learning and Kann et al. (2018), who refer to the approach as "multi-task learning", found it superior to Silfverberg et al. (2017)'s strategy of replacing common substrings in the test data with random strings from the training data. Simplistically, we might assume that copying "base forms" (e.g. "run") would work well for morphologically simple languages that add few affixes to the base form, but the foundation of morphological analysis is the systematic patterns of inflection which means that all languages tend to repeat identical or very similar substrings across morphologically related forms.

5 Computational Morphology for Language Documentation and Description

This section discusses how the computational approaches discussed in this paper could be applied even further to low-resource languages. It looks specifically at how computational methods might be integrated into the creation of new linguistic resources that are commonly created during intermediate stages of documentary and descriptive linguistic fieldwork. This overview indicates

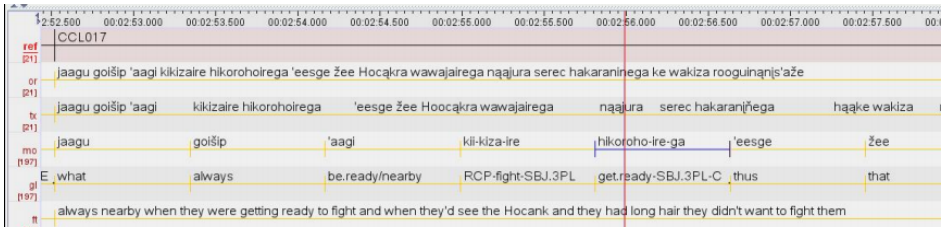


Figure 12: Interlinearization in ELAN.

that the simplest and most effective way for LDD to benefit from computational morphology is to leverage manually glossed texts that it already produces.

5.1 Interlinearization: A Bottleneck

Although computational linguistics has a solid history in morphology and recent impressive success in low-resource contexts, documentary and descriptive linguistics has witnessed little evidence of this. Few computational models have been applied to basic morphological analysis and annotation. Annotation stands as one of several bottlenecks in production of accessible endangered language data (Holton et al., 2017). This bottleneck is doubly unfortunate because such data is a vital resource to both linguistics and NLP.

In a typical LDD workflow, morphological analysis and annotation takes center stage as soon as new language data has been recorded and transcribed.[6] The analysis and annotation happens during the process of interlinearization. Narrowly defined, interlinearization is annotation that marks boundaries between morphemes (segmentation), labels morphemes with their morphosyntactic or lexical meaning (glossing), and provides a rough equivalent of each sentence in a language of wider communication (free translation). The outcome is often referred to as interlinear glossed texts or IGT. Interlinearization It stands on the fuzzy line between documentary linguistics and descriptive linguistics because it goes beyond mere documentation of a language but still serves as a "preprocessing step" (Moon et al., 2009) to deeper linguistic analysis. It lays the foundation for the fuller descriptions that are published in dictionaries and reference grammars.

NLP assistance to annotators is limited, and machine learning assistance is not readily available

[6]Transcription is the first major bottleneck in data production. Fortunately, machine learning is already being applied to this step, cf. (Foley et al., 2018).

to those who do not have computer programming skills. Linguistic software tools that aid interlinearization only incorporate rule-based computational models. ELAN, shown in Figure 5.1[7], and FLEx, shown in Figure 13[8] are two popular tools. Neither use machine learning, although they do present automated suggestions for segmentation and glossing. However, the suggestions are only given if a word form is identical to one that was previously annotated or if the user has constructed the rules for the morphological analyzer.

This situation persists, even though it has been known for some time that machine learning can be effectively applied to the LDD tasks (Baldridge and Palmer, 2009; Palmer, 2009; Duong, 2017). Interlinearization is performed primarily by hand, and while human analysis is the most reliable because a human is much better able to interpret contextual cues, once the initial analysis is complete, manual annotation is extremely inefficient (Baldridge and Osborne, 2008; Baldridge and Palmer, 2009; Palmer, 2009). It is repetitive and monotonous, and, therefore, prone to typos and inconsistencies when done manually. Manual work is costly, requiring money and time to hire and train annotators (Duong, 2017; He et al., 2016). It is time-consuming; even in computational linguistics it took three years to annotate just one layer of the Penn Treebank (Taylor et al., 2003). A documentary field project may record several hours of valuable and endangered data, but interlinearization is rarely completed within funding and time limits.

5.2 Immediate Possibilities

The bottleneck of interlinearization could be addressed right now by integrating some of the machine learning models and techniques described in this paper. By training on manually produced IGT, supervised learning models can speed interlinearization and support morphological description beyond it, such as inflectional paradigm induction. Simply splicing together some of the computational approaches described in this paper could reduce the time cost by as much as 90% (Felt, 2012; Moeller and Hulden, 2018).

[7] Source: Bouda et al. (2012)
[8] Source: http://software.sil.org/fieldworks/resources/tutorial/interlinearize-texts/

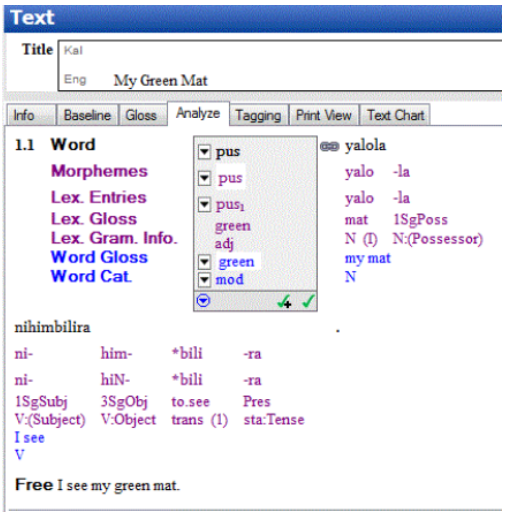


Figure 13: Interlinearization in Fieldworks Language Explorer (FLEx).

Morphological analysis and annotation. There many ways to integrate machine learning into the annotation stage of interlinearization. A simple one involves a CRF that is trained with simple, cross-linguistic features can be trained on previously segmented data. Such a model has achieved over 93% accuracy using only 3,000 training words to segment and label the closed class of affixes in Lezgi (Moeller and Hulden, 2018) (see section 3.3). A Transformer might outperform the CRF even in very low data settings. It would be would be worth comparing both. The models would certainly perform better if trained only on the closed class of affixes. On the open class of roots, accuracy would be lower, but could be improved with some of the augmentation techniques described in section 4. For example, existing dictionaries or lexicons or an FST can be exploited to increase training data for a neural model as Moeller et al. (2018) and Moeller et al. (2019) did (see section 3.4). If these resources are insufficient, word lists from languages with heavy borrowings can be used as Baumann and Pierrehumbert (2014) did (see section 4.2).

Paradigm induction. Even with available IGT the approaches described in this paper regarding paradigm induction (see section 2.2) can serve as a jump start for the next stages of LDD. Specifically, IGT could be leveraged for automating initial identification of morphological patterns and classifying those patterns as inflectional classes. For example, a task, illustrated in Figure 14 which we can call **IGT-to-paradigms** (IGT2P), builds off existing *morphological inflection* tasks

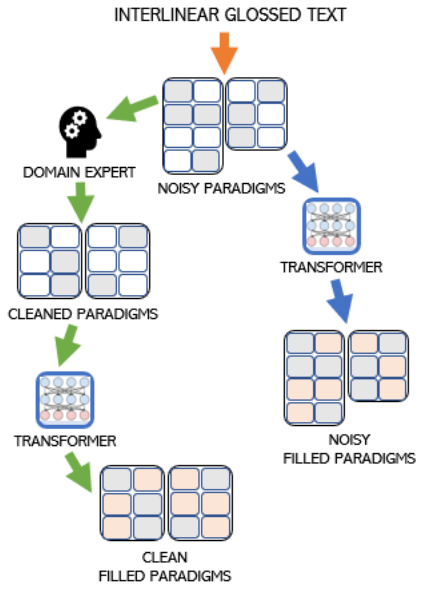


Figure 14: IGT2P. Inflected word forms attested in interlinear glossed texts (IGT) train transformer encoder-decoder to generalize morphological paradigmatic patterns and generate word forms when given known morphosyntatic features of missing paradigm cells. Noisy paradigms are automatically constructed from IGT and a language expert creates "cleaned" paradigms. Both sets are tested on the same missing word forms and the results are compared.

(Yarowsky and Wicentowski, 2000; Faruqui et al., 2016). Working with LDD data gives it three unique aspects: (1) inflected forms extracted from IGTs are noisier than curated training data, (2) since lemmas are not explicitly identified in IGTs, systems cannot be trained on typical lemma-to-form mappings and, instead, must be trained on form-to-form mappings, and (3) part-of-speech (POS) tags are often unavailable in IGTs. IGT2P can thus be seen as a noisy version of morphological *re*inflection (Cotterell et al., 2016a), but without explicit POS information. One experiment has shown that this IGT2P is successful for generating unseen inflected forms. This is along the line of the PCFP illustrated in Figure 2.

Some results from an IGT2P experiment are shown in Table 5.2. IGT2P could assist the initial analysis of morphological patterns in IGT. By quickly learning morphological patterns from word forms attested in IGTs, IGT2P generates forms that fill empty cells in a lemma's paradigm. Since IGTs are unlikely to contain complete paradigms of lemmas, an accompanying step in fieldwork is that of elicitation of inflectional paradigms for selected lemmas. Presenting candidate words to

	T	+aug	+uninfl	+both	mono	+aug	+uninfl	+both
arp clean	**62.08**	61.39	61.58	60.78	15.93	15.75	15.58	15.94
arp noisy	57.77	57.64	*58.04*	57.51	14.51	14.64	14.52	14.69
ddo clean	65.38	**66.53**	65.19	65.42	59.9	60.87	59.53	60.64
ddo noisy	63.54	63.95	62.89	*64.04*	59.12	58.66	57.87	57.97
lez clean	46.59	32.95	46.59	**48.86**	32.95	35.23	31.82	31.82
lez noisy	*35.23*	29.55	32.95	27.27	30.68	28.41	20.45	31.82
mni clean	30.63	30.87	31.81	**32.04**	23.24	25.7	21.95	24.77
mni noisy	21.48	*22.3*	21.6	21.83	18.78	18.31	19.37	20.31
ntu clean	**53.18**	46.82	49.15	48.52	29.66	33.9	28.18	33.05
ntu noisy	36.86	45.55	45.34	*45.76*	31.99	33.69	31.78	30.93

Table 2: Results of reinflection task for transformer model (T) and the LSTM seq2seq model with exact hard monotonic attention (mono) with artificial data augmention (+aug), addition of unannotated/uninflected word forms (+uninfl) and both together. Boldface indicates best result; italics indicate best result on noisy paradigms.

a native speaker for acceptance or rejection is often easier than asking the speaker to grasp the abstract concept of a paradigm and to generate the missing cells in a table. With the help of IGT2P, linguists could use the machine-generated word forms to support this elicitation process. IGT2P then becomes a tool for the discovery of morphological patterns in under-described and endangered languages.

5.3 Next step

The plan in the previous subsection is mostly limited to the models and methods described in this paper. These are essential experiments in integration of machine learning, but once the best models and methods are identified, adding a new computational method would be a boon to documentary and descriptive linguists (and to NLP by increasing available training data in return). This method is called active learning.

Baldridge and Palmer (2009) propose two stratagems to improve and speed LDD data production: reduce the cost of annotating new data or else get more miles out of previously annotated data. Crowdsourcing is an example of the first strategem but has proved less effective in low-resource contexts (Bird et al., 2014; Bettinson and Bird, 2017). For the second strategem, Baldridge et al.

recommend active learning.

Active learning is a machine learning technique that is ideal for situations where raw data is abundant but manual annotation is expensive. It is conducted as an iterative cycle of annotating data then (re-)training a supervised model. First the model is trained and tested, then run on unseen data. Another computation model directs human annotators towards a certain number of the most informative instances in that data. Informative instances are those that are most likely to improve the model upon re-training. For example, these might be word forms that are most dissimilar to any in the training data. Annotators then manually vet and correct the model's predictions for these instances.[9] The model is re-trained with the corrected annotations included the training data. The cycle continues until desired accuracy is achieved or a point of diminishing returns has been reached. Any remaining bit of incorrect predictiosn will be vetted and corrected by hand.

Experiments in computer-aided machine translation (Kothur et al., 2018) and interlinearizion (Palmer, 2009; Palmer et al., 2009; Palmer et al., 2010) indicate that active learning is a very promising route for LDD. Active learning applied with computational morphological models to an Uspanteko [usp] documentary corpus gave a significant boost in the models' accuracy (Palmer, 2009). At the same time, the annotators' productivity was increased and the time cost of annotation greatly reduced.

The iterative nature of active learning calls for human interaction. The development of well-designed graphical user interfaces is vital. Unfortunately, "ordinary working linguists" and the average native speaker annotator would not find the typical model's output easy to work with, and may actually find them intimidating. This gap of skills and knowledge must be bridged. The value of such an interface can be seen by the one provided by ELPIS (Foley et al., 2018), a wonderful application of machine learning integrated into the transcription bottleneck. However, it should not only walk annotators through the steps of annotating data and training a model, but its graphics would draw their attention to samples that most need to be vetted or corrected and recommend when re-training should take place. Instead, it leaves the annotator to examine each and every

[9]This step is done manually, but Rocio et al. (2007) found that even this can be semi-automated if the model's patterns of errors are identified.

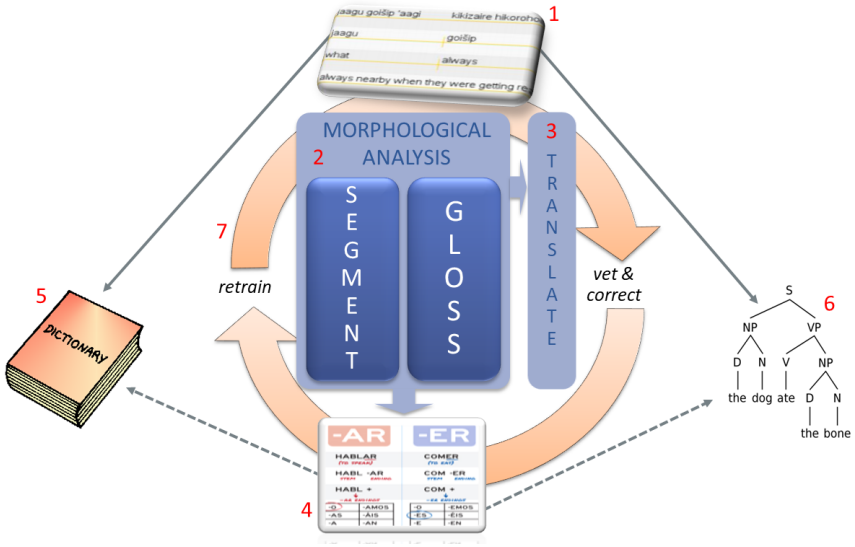


Figure 15: A practical application of computational morphology to language documentation and description would focus on addressing the time costs of interlinearization (1). Incorporating computational models at the segmentation and glossing steps (2) could feed into machine translation models (3), which would help complete interlinearization, and then into paradigm induction models which could provide working hypotheses of inflectional patterns and classes (4). This would support the development of dictionaries (5) and grammatical descriptions (6). Even with very limited initial supervision, the models can continue to improve if a cycle of active learning (7) is introduced at points in this workflow. This means annotators would vet and correct the most informative of the models' prediction and that information would be used to retrain the models.

symbol to find and correct errors.

Ideally, linguists will someday spend minimal time on manual interlinearization and other linguistic annotation, followed by a few cycles of training a supervised machine learning model, vetting and correcting its results, and retraining the model until it achieves extremely high accuracy. The primary question is not whether a LDD project annotated sufficient data, but how many rounds of re-training and manual correction will be required before it becomes more expensive than finishing the annotation by hand.

6 Conclusion

All computational morphological models—finite state transducers, unsupervised, supervised feature-based or neural networks, and semi-supervised—need data. Unsupervised models achieve some success with unannotated data, but accuracy is much higher with supervised and semi-

supervised machine learning which require annotated data. Despite the importance of annotated data, the computational linguistics literature repeatedly returns to the unfortunate reality that "annotating sufficient data...is expensive" (Cotterell and Heigold, 2017, p. 1954). This has retarded the development of NLP and its potential contribution to linguistic science. The literature adds another perceived inconvenience: that annotation "relies on linguistic expertise" (*ibid*). That is, a language expert—native speaker or linguist—must be involved to produce gold standard annotations necessary for training and evaluating machine learning models or for building finite state machines.

Morphological annotation of texts (interlinearization) also stands as a bottleneck to language documentation and language description (Bird et al., 2015; Bettinson and Bird, 2017; Holton et al., 2017). It is not, however, the "inconvenient" reliance on people who actually know the language. Making the most of native speaker knowledge has long been an integral part of linguistics and language documentation literature is brimming with proven methods for doing so. What is remains is the need to reduce the cost of annotation and speed its production even while existing resources are few.

Fortunately, computational approaches are being increasingly applied to low-resource contexts and are achieving reasonable success. This paper compared various computational approaches to morphological analysis. It described much of what has already been accomplished in low-resource contexts. Finally, it presented how this trend has the potential to address practical needs and bottlenecks in the language documentation and language description workflow. It outlined a plan to incorporate existing models and then suggested that active learning and well-designed user interfaces are the strategic next steps to speed and improve the production of annotated data.

Bridging this technological gap between computational morphology and language documentation and description will undoubtedly open new questions for future research. For NLP, the questions are likely to focus on how models and techniques should adjust so that they perform optimally across a wide range of languages. For linguistics, a big question is how machine learning will affect best practices in field methods. Future work must seek an optimal balance between NLP

goals and the priorities of field linguists, as well as the priorities and needs of the communities who speak low-resource and often endangered languages.

References

Farrell Ackerman and Robert Malouf. 2009. Parts and wholes: Patterns of relatedness in complex morphological systems and why they matter. In J. P. Blevins and J. Blevins, editors, *Analogy in grammar: Form and acquisition*, pages 54–82. Oxford University Press, Oxford.

Malin Ahlberg, Markus Forsberg, and Mans Hulden. 2014. Semi-supervised learning of morphological paradigms and lexicons. In *Proceedings of 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 569–578. Association for Computational Linguistics.

Malin Ahlberg, Markus Forsberg, and Mans Hulden. 2015. Paradigm classification in supervised learning of morphology. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1024–1029. Association for Computational Linguistics.

Nicholas Andrews, Mark Dredze, Benjamin Van Durme, and Jason Eisner. 2017. Bayesian modeling of lexical resources for low-resource settings. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1:1029–1039.

Peter K. Austin. 2014. Language documentation in the 21st century. *JournaLIPP*, 0(3):57–71.

Jason Baldridge and Miles Osborne. 2008. Active learning and logarithmic opinion pools for hpsg parse selection. *Natural Language Engineering*, 14(2):191–222.

Jason Baldridge and Alexis Palmer. 2009. How well does active learning actually work? time-based evaluation of cost-reduction strategies for language documentation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 296–305.

Marco Baroni, Johannes Matiasek, and Harald Trost. 2002. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. In *Proceedings of the ACL-02 workshop on Morphological and phonological learning-Volume 6*, pages 48–57. Association for Computational Linguistics.

Peter Baumann and Janet Pierrehumbert. 2014. Using resource-rich languages to improve morphological analysis of under-resourced languages. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Kenneth R Beesley and Lauri Karttunen. 2003. *Finite-State Morphology: Xerox Tools and Techniques*. Xerox Corporation.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 31–39.

Mat Bettinson and Steven Bird. 2017. Developing a suite of mobile applications for collaborative language documentation. In *Workshop on Computational Methods for Endangered Languages*.

Steven Bird, Florian Hanke, Oliver Adams, and Lee Haejoong. 2014. Aikuma: A mobile app for collaborative language documentation. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 1–5. Association for Computational Linguistics.

Steven Bird, David Chiang, Friedel Frowein, Florian Hanke, and Ashish Vaswani. 2015. Documentary linguistics and computational linguistics: A response to brooks. *Language Documentation and Consveration*.

Steven Bird. 2009. Natural language processing and linguistic fieldwork. *Computational Linguistics*, 35(3):469–474.

James P Blevins. 2006. Word-based morphology. *Journal of Linguistics*, pages 531–573.

Brenda H. Boerger, Sarah Ruth Moeller, Will Reiman, and Stephen Self. 2016. *Language and Culture Documentation Manual*. Leanpub.

Peter Bouda, Johannes Helmbrecht, and Ludwig-Maximilians-Universität München. 2012. From corpus to grammar: how DOBES corpora can be exploited for descriptive linguistics. In Sebastian Nordhoff, editor, *Electronic Grammaticography*, number 4 in Language Documentation & Conservation Special Publication, pages 129–159. University of Hawaii.

Jan Buys and Jan A. Botha. 2016. Cross-lingual morphological tagging for low-resource languages. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Erwin Chan. 2006. Learning probabilistic paradigms for morphology in a latent class model. In *Proceedings of the Eighth Meeting of the ACL Special Interest Group on Computational Phonology and Morphology*, SIGPHON '06, pages 69–78. Association for Computational Linguistics.

Noam Chomsky and Morris Halle. 1968. *The sound patterns of English*. ERIC.

Shay B. Cohen and Noah A. Smith. 2007. Joint morphological and syntactic disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 208–217. Association for Computational Linguistics.

Ryan Cotterell and Georg Heigold. 2017. Cross-lingual character-level neural morphological tagging. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 748–759.

Ryan Cotterell, Thomas Müller, Alexander M. Fraser, and Hinrich Schütze. 2015. Labeled morphological segmentation with semi-markov models. In *CoNLL*, pages 164–174.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016a. The SIGMORPHON 2016 shared Task—Morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22, Berlin, Germany, August. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016b. The SIGMORPHON 2016 shared task—morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sebastian Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27. Association for Computational Linguistics.

Andrew Cowell and Alonzo Moss. 2008. *The Arapaho Language*. University Press of Colorado.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 workshop on Morphological and phonological learning-Volume 6*, pages 21–30. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2005. *Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0*. Helsinki University of Technology.

Mathias Creutz and Krista Lagus. 2007a. Inducing the morphological lexicon of a natural language from unannotated text. In *In Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05)*.

Mathias Creutz and Krista Lagus. 2007b. Unsupervised models for morpheme segmentation and morphology learning. *ACM Trans. Speech Lang. Process.*, 4(1):3:1–3:34.

Mathias Creutz. 2003. Unsupervised segmentation of words using prior distributions of morph length and frequency. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 280–287. Association for Computational Linguistics.

Marie-Catherine De Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal stanford dependencies: A cross-linguistic typology. In *LREC*, volume 14, pages 4585–92.

Markus Dreyer and Jason Eisner. 2011. Discovering morphological paradigms from plain text using a dirichlet process mixture model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 616–627. Association for Computational Linguistics.

Gregory Druck, Gideon Mann, and Andrew Mccallum. 2007. Reducing annotation effort using generalized expectation criteria. Technical report, University of Massachusetts Amherst, Dept. of Computer Science.

Long Duong. 2017. *Natural language processing for resource-poor languages*. Phd thesis, University of Melbourne.

Greg Durrett and John DeNero. 2013. Supervised learning of complete morphological paradigms. In *Proceedings of NAACL-HLT*, pages 1185–1195. Association for Computational Linguistics.

Grégoire Détrez and Aarne Ranta. 2012. Smart paradigms and the predictability and complexity of inflectional morphology. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 645–653. Association for Computational Linguistics.

David M. Eberhard, Gary F. Simons, and Charles D. Fennig, editors. 2020. *Ethnologue: Languages of the World*. SIL International, Dallas, Texas, twenty-third edition.

Jeffrey L. Elman. 1991. Distributed representations, simple recurrent networks, and grammatical structure. *MACHINE LEARNING*, 7(2):195–225.

Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2016. Morphological inflection generation using character sequence to sequence learning. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 634–643, San Diego, California, June. Association for Computational Linguistics.

Paul Felt. 2012. *Improving the Effectiveness of Machine-Assisted Annotation*. Phd thesis.

Raphael Finkel and Gregory Stump. 2007. Principal parts and morphological typology. *Morphology*, 17(1):39–75.

Ben Foley, Josh Arnold, Rolando Coto-Solano, Gautier Durantin, T. Mark Ellison, Daan van Esch, Scott Heath, František Kratochvíl, Zara Maxwell-Smith, David Nash, Ola Olsson, Mark Richards, Nay San, Hywel Stoakes, Nick Thieberger, and Janet Wiles. 2018. Building speech recognition systems for language documentation: The CoEDL endangered language pipeline and inference system. In *Proceedings of the 6th International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU 2018)*.

Markus Forsberg and Mans Hulden. 2016. Learning transducer models for morphological analysis from example inflections. In *Proceedings of the SIGFSM Workshop on Statistical NLP and Weighted Automata*, pages 42–50. Association for Computational Linguistics.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. One sense per discourse. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*. Association for Computational Linguistics.

John Goldsmith, Jackson Lee, and Aris Xanthos. 2017. Computational learning of morphology. *Annual Review*, 3.

John Goldsmith. 2000. Linguistica: An automatic morphological analyzer. In *Meeting of the Chicago Linguistic Society*, volume 1.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational linguistics*, 27(2):153–198.

Michael Wayne Goodman. 2013. Generation of machine-readable morphological rules from human-readable input. *UW Working Papers in Linguistics*, 30.

Harald Hammarström and Lars Borin. 2011. Unsupervised learning of morphology. *Computational Linguistics*, 37(2):309–350.

Zellig Harris. 1955. From phoneme to morpheme. *Language*, 31(2):190–222.

Zellig Harris. 1967. Morpheme boundaries within words: Report on a computer test. In *Transformations and Discourse Analysis Papers*, volume 73. Department of Linguistics, University of Pennsylvania.

Luheng He, Julian Michael, Mike Lewis, and Luke Zettlemoyer. 2016. Human-in-the-loop parsing. In *EMNLP*, pages 2337–2342.

Georg Heigold, Guenter Neumann, and Josef van Genabith. 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.

Nikolaus P. Himmelmann. 1998. Documentary and descriptive linguistics. *Linguistics*, 36:161–195.

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

Gary Holton, Kavon Hooshiar, and Nicholas Thieberger. 2017. Developing collection management tools to create more robust and reliable linguistic data. In *Workshop on Computational Methods for Endangered Languages*.

Mans Hulden. 2009. *Finite-state Machine Construction Methods and Algorithms for Phonology and Morphology*. Phd thesis, University of Arizona.

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2016. Neural multi-source morphological reinflection. *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*.

Katharina Kann, Jesus Manuel Mager Hois, Ivan Vladimir Meza Ruiz, and Hinrich Schütze. 2018. Fortification of neural morphological segmentation models for polysynthetic minimal-resource languages. 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 47–57. Association for Computational Linguistics.

Ronald M Kaplan and Martin Kay. 1981. Phonological rules and finite-state transducers. In *Linguistic Society of America Meeting Handbook, Fifty-Sixth Annual Meeting*, pages 27–30.

Lauri Karttunen and Kenneth R. Beesley. 2005. Twenty-five years of finite-state morphology. In *Inquiries Into Words, a Festschrift for Kimmo Koskenniemi on his 60th Birthday*, pages 71–83. CSLI publications.

Amit Kirschenbaum, Peter Wittenburg, and Gerhard Heyer. 2012. Unsupervised morphological analysis of small corpora: First experiments with kilivila. *Language Documentation & Conservation Special Publication*, 3:25–31.

Oskar Kohonen, Sami Virpioja, and Krista Lagus. 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. Association for Computational Linguistics.

Kimmo Koskenniemi. 1983. Two-level model for morphological analysis. In *IJCAI*, volume 83, pages 683–685.

Sachith Sri Ram Kothur, Rebecca Knowles, and Philipp Koehn. 2018. Document-level adaptation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 64–73. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando C N Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289.

Jackson Lee and John Goldsmith. 2016. Linguistica 5: Unsupervised learning of linguistic structure. In *Proceedings of NAACL-HLT 2016 (Demonstrations)*, pages 222–26. Association for Computational Linguistics.

Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2011. Modeling syntactic context improves morphological segmentation. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 1–9. Association for Computational Linguistics.

Christian Lehmann. 1999. Documentation of endangered languages: A priority task for linguistics. In *Arbeitspapiere des Seminars fur Sprachwissenschaft der Universitat Eurfurt*, volume 1. Seminar fur Sprachwissenschaft der Universitat.

Ling Liu and Mans Hulden. 2020. Leveraging Principal Parts for Morphological Inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 153–161. Association for Computational Linguistics, July.

Ling Liu, Ilamvazhuthy Subbiah, Adam Wiemerslage, Jonathan Lilley, and Sarah Moeller. 2018. Morphological reinflection in context: CU boulder's submission to CoNLL-SIGMORPHON 2018 shared task. In *Proceedings of the CoNLL SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 86–92. Association for Computational Linguistics.

Friederike Lupke. 2010. Data collection methods for field-based language documentation. *Language Documentation and Description*, 7:55–104.

Peter Makarov and Simon Clematide. 2018. UZH at CoNLL-SIGMORPHON 2018 shared task on universal morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 69–75. Association for Computational Linguistics.

Peter Makarov, Tatiana Ruzsics, and Simon Clematide. 2017. Align and copy: UZH at SIGMORPHON 2017 shared task for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 49–57. Association for Computational Linguistics.

Robert Malouf. 2016. Generating morphological paradigms with a recurrent neural network. *San Diego Linguistic Papers*, 6:122–129.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Crosslinguality and context in morphology. In *Proceedings of the 16th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Florence, Italy. Association for Computational Linguistics.

Sarah Moeller and Mans Hulden. 2018. Automatic glossing in a low-resource setting for language documentation. In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, pages 84–93. Association for Computational Linguistics.

Sarah Moeller, Ghazaleh Kazeminejad, Andrew Cowell, and Mans Hulden. 2018. A neural morphological analyzer for arapaho verbs learned from a finite state transducer. In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, pages 12–20. Association for Computational Linguistics.

Sarah Moeller, Ghazaleh Kazeminejad, Andrew Cowell, and Mans Hulden. 2019. Improving low-resource morphological learning with intermediate forms from finite state transducers. In *Proceedings of the Workshop on Computational Methods for Endangered Languages*.

Christian Monson, Alon Lavie, Jaime Carbonell, and Lori Levin. 2004. Unsupervised induction of natural language morphology inflection classes. In *Proceedings of the 7th Meeting of the ACL Special Interest Group in Computational Phonology: Current Themes in Computational Phonology and Morphology*, pages 52–61. Association for Computational Linguistics.

Christian Monson, Jaime Carbonell, Alon Lavie, and Lori Levin. 2007a. ParaMor: Finding paradigms across morphology. In *Advances in Multilingual and Multimodal Information Retrieval*, Lecture Notes in Computer Science, pages 900–907. Springer, Berlin, Heidelberg.

Christian Monson, Jaime Carbonell, Alon Lavie, and Lori Levin. 2007b. Paramor: Minimally supervised induction of paradigm structure and morphological analysis. In *Proceedings of Ninth Meeting of the ACL Special Interest Group in Computational Morphology and Phonology*, pages 117–125. Association for Computational Linguistics.

Christian Monson, Alon Lavie, Jaime Carbonell, and Lori Levin. 2008. Evaluating an agglutinative segmentation model for paramor. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology*, pages 49–58. Association for Computational Linguistics.

Taesun Moon, Katrin Erk, and Jason Baldridge. 2009. Unsupervised morphological segmentation and clustering with document boundaries. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 668–677. Association for Computational Linguistics.

Thomas Müller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 322–332.

Garrett Nicolai and Grzegorz Kondrak. 2017. Morphological analysis without expert annotation. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 2:211–216.

Alexis Palmer, Taesun Moon, and Jason Baldridge. 2009. Evaluating automation strategies in language documentation. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, HLT '09, pages 36–44. Association for Computational Linguistics.

Alexis Palmer, Taesun Moon, Jason Baldridge, Katrin Erk, Eric Campbell, and Telma Can. 2010. Computational strategies for reducing annotation effort in language documentation. *Linguistic Issues in Language Technology*, 3(4):1–42.

Alexis Mary Palmer. 2009. *Semi-automated annotation and active learning for language documentation*. Phd thesis, University of Texas at Austin.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 209–217. Association for Computational Linguistics.

Brian Roark and Richard William Sproat. 2007. *Computational approaches to morphology and syntax*. Oxford University Press.

Vitor Rocio, Joaquim Silva, and Gabriel Lopes. 2007. Detection of strange and wrong automatic part-of-speech tagging. In *Proceedings of the Aritficial Intelligence 13th Portuguese Conference on Progress in Artificial Intelligence*, volume 4874, pages 683–690. Springer Berlin Heidelberg.

Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja, and Mikko Kurimo. 2013. Supervised morphological segmentation in a low-resource learning setting using conditional random fields. In *CoNLL*, pages 29–37.

Teemu Ruokolainen, Oskar Kohonen, Kairit Sirts, Stig-Arne Grönroos, Mikko Kurimo, and Sami Virpioja. 2016. A comparative study of minimally supervised morphological segmentation. *Computational Linguistics*, 42(1):91–120.

Patrick Schone and Daniel Jurafsky. 2001. Knowledge-free induction of inflectional morphologies. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 1–9. Association for Computational Linguistics.

Burr Settles. 2010. Active learning literature survey. *University of Wisconsin, Madison*, 52(55):11.

Miikka Silfverberg and Mans Hulden. 2018. An encoder-decoder approach to the paradigm cell filling problem. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2883–2889. Association for Computational Linguistics.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99.

Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. In *ACL*, pages 737–745.

Radu Soricut and Franz Och. 2015. Unsupervised morphology induction using word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1627–1637.

Akhilesh Sudhakar and Anil Kumar Singh. 2017. Experiments on morphological reinflection: CoNLL-2017 shared task. *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 71–78.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3104–3112.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The penn treebank: An overview. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, Text, Speech and Language Technology, pages 5–22. Springer Netherlands.

Nick Thieberger, Anna Margetts, Stephen Morey, and Simon Musgrave. 2016. Assessing annotated corpora as research output. *Australian Journal of Linguistics*, 36(1):1–21.

Nicholas Thieberger. 2012. Using language documentation data in a broader context. *Language Documentation & Conservation Special Publication*, 3:129–134.

Rosa Vallejos. 2014. Integrating language documentation, language preservation, and linguistic research: Working with the kokamas from the amazon. *Language Documentation & Conservation*, 8:38–65.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1:2016–2027.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6000–6010, Long Beach, California, USA. Curran Associates Inc.

Sami Virpioja, Ville Turunen, Sebastian Spiegler, Oskar Kohonen, and Mikko Kurimo. 2011. Empirical comparison of evaluation methods for unsupervised learning of morphology. *TAL*, 52(2):45–90.

Linlin Wang, Zhu Cao, Yu Xia, and Gerard de Melo. 2016. Morphological segmentation with window LSTM neural networks. In *AAAI*, pages 2842–2848.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2020. Applying the transformer to character-level transduction. *arXiv:2005.10213 [cs.CL]*, May.

David Yarowsky and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 207–216, Hong Kong, October. Association for Computational Linguistics.