Unsupervised Morphological Paradigm Completion

Huiming Jin, Liwei Cai, Yihui Peng, Chen Xia
Language Technologies Institute
Carnegie Mellon University
{huimingj,liweicai,yihuip,chenxia}@cs.cmu.edu

Arya D. McCarthy
Johns Hopkins University
arya@jhu.edu

Katharina Kann
University of Colorado
katharina.kann@colorado.edu

Abstract

We propose the task of *unsupervised morphological paradigm completion*. Given only raw text and a lemma list, the task consists of generating the morphological paradigms, i.e., all inflected forms, of the lemmas. From a natural language processing (NLP) perspective, this is a challenging unsupervised task, and high-performing systems have the potential to improve tools for low-resource languages or to assist linguistic annotators. From a cognitive science perspective, this can shed light on how children acquire morphological knowledge. We further introduce a system for the task, which generates morphological paradigms via the following steps: (i) EDIT TREE retrieval, (ii) additional lemma retrieval, (iii) paradigm size discovery, and (iv) inflection generation. We perform an evaluation on 14 typologically diverse languages. Our system outperforms trivial baselines with ease and, for some languages, even obtains a higher accuracy than minimally supervised systems.[1]
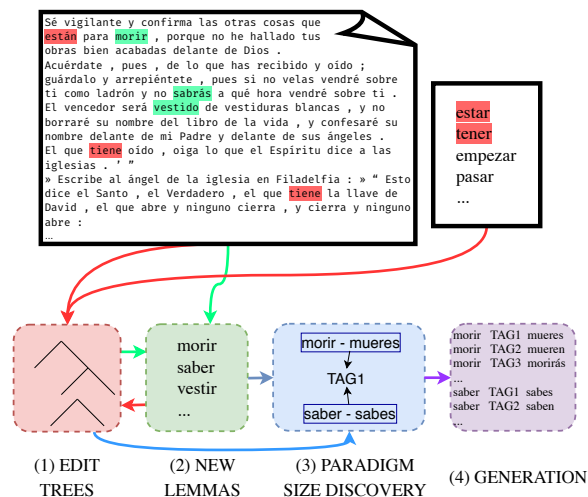
Figure 1: Our unsupervised paradigm completion system, which takes raw text and a lemma list as inputs. We describe it in detail in §4.

1 Introduction

Morphologically rich languages express syntactic and semantic properties—like *tense* or *case*—of words through inflection, i.e., changes to the surface forms of the words. The set of all inflected forms of a *lemma*—the canonical form—is called its *paradigm*. While English does not manifest a rich inflectional morphology, Polish verbs have around a hundred different forms (Sadowska, 2012), and Archi paradigms, an extreme example, can have over 1.5 million slots (Kibrik, 1977).

Morphologically rich languages constitute a challenge for natural language processing (NLP) systems: because each lemma can take on a variety of surface forms, the frequency of each individual inflected word decreases drastically. This yields problems for speech recognition (Creutz

et al., 2007), parsing (Seeker and Çetinoğlu, 2015), and keyword spotting (Narasimhan et al., 2014), *inter alia*. For unsupervised machine translation, Guzmán et al. (2019) encounter difficulties when translating into the morphologically rich languages Nepalese and Sinhalese.

Children acquire morphological knowledge from raw utterances and, in particular, without access to explicit morphological information (Berko, 1958). Do they have an innate capacity that enables them to learn a language's morphology? Or can morphology be learned in an unsupervised fashion? This question—in addition to practical considerations like benefits for the aforementioned NLP tasks—has motivated work on unsupervised morphological analyses (Goldsmith, 2001; Creutz, 2003). To the best of our knowledge, no previous work has considered unsupervised morphological *generation*.[2] However, over the last few years, there has

---

[1] Our implementation is available under https://github.com/cai-lw/morpho-baseline.

[2] Kann et al. (2017), which performs a zero-shot inflection experiment, uses prior information about paradigm size and related languages and, thus, cannot draw the same conclusions.

been a lot of progress on morphological generation tasks with limited amounts of supervision, in particular on morphological inflection (Cotterell et al., 2018) and paradigm completion (Kann and Schütze, 2018), which can potentially be leveraged for unsupervised solutions.

Here, we fill the gap between unsupervised morphological analysis and morphological generation with limited training data by proposing the task of *unsupervised morphological paradigm completion*. That is, we aim to construct and fill inflection tables exclusively from raw text and a lemma list for a known part of speech (POS), in a situation similar to those encountered by field linguists. We further present a system for the task (see Figure 1) which employs state-of-the-art methods common in NLP and computational morphology. It performs the following four steps: (i) EDIT TREE (Chrupała, 2008) retrieval (§4.1), (ii) additional lemma retrieval (§4.2), (iii) paradigm size discovery using distributional information (§4.3), and (iv) inflection generation (§4.4).

To evaluate our approach, we design a metric for unsupervised paradigm completion, best-match accuracy (§5.4), and experiment on 14 languages from 7 families. As we are tackling a novel task with no baselines in the NLP literature, we perform an extensive ablation study to demonstrate the importance of all steps in our pipeline. We further show that our system outperforms trivial baselines and, for some languages, even obtains higher accuracy than a minimally supervised system.

2 Related Work

Morphological Generation Versions of our task with varying degrees of supervision—though never totally unsupervised—have been explored in the past. Yarowsky and Wicentowski (2000) is the previous work most similar to ours. They also assume raw text and a word list as input, but additionally require knowledge of a language's consonants and vowels, as well as canonical suffixes for each part of speech. Dreyer and Eisner (2011) assume access to seed paradigms to discover paradigms in an empirical Bayes framework. Ahlberg et al. (2015) and Hulden et al. (2014) combine information about paradigms and word frequency from corpora to perform semi-supervised paradigm completion. Our work differs from them in that we do *not* assume any gold paradigms to be given.

Durrett and DeNero (2013), Nicolai et al. (2015), and Faruqui et al. (2016) explore a fully supervised approach, learning morphological paradigms from large annotated inflection tables. This framework has evolved into the SIGMORPHON shared tasks on morphological inflection (Cotterell et al., 2016), which have sparked further interest in morphological generation (Kann and Schütze, 2016; Aharoni and Goldberg, 2017; Bergmanis et al., 2017; Makarov et al., 2017; Zhou and Neubig, 2017; Kann and Schütze, 2018). We integrate two systems (Cotterell et al., 2017; Makarov and Clematide, 2018b) produced for SIGMORPHON shared tasks into our framework for unsupervised morphological paradigm completion.

Morphological Analysis Most research on unsupervised systems for morphology aims at developing approaches to segment words into their smallest meaning-bearing units, called *morphemes* (Goldsmith, 2001; Creutz, 2003; Creutz and Lagus, 2007; Snyder and Barzilay, 2008). Unsupervised morphological paradigm completion differs from segmentation in that, besides capturing how morphology is reflected in the word form, it also requires correctly clustering transformations into paradigm slots as well as generating unobserved forms. The model by Xu et al. (2018) recovers something akin to morphological paradigms. However, those paradigms are a means to a segmentation end, and Xu et al. (2018) do not explicitly model information about the paradigm size as required for our task.

Other unsupervised approaches to learning morphological analysis and generation rely on projections between word embeddings (Soricut and Och, 2015; Narasimhan et al., 2015); however, these approaches rely on billions of words to train embeddings; at a minimum, Narasimhan et al. (2015) use 129 million word tokens of English Wikipedia. As we will describe later on (§5.1), we, in contrast, are concerned with the setting with mere thousands of sentences.

For a detailed survey of unsupervised approaches to problems in morphology, we refer the reader to Hammarström and Borin (2011).

SIGMORPHON 2020: Unsupervised Morphological Paradigm Completion After multiple shared tasks on morphological inflection starting with Cotterell et al. (2016), in 2020, SIGMORPHON (the ACL special interest group on computational morphology and phonology) is organizing

its first shared task on unsupervised morphological paradigm completion.[3] The system presented here is the official shared task baseline system. The first other approach applicable to this shared task has been developed by Erdmann et al. (2020). Their pipeline system is similar in spirit to ours, but the individual components are different, e.g., a transformer model (Vaswani et al., 2017) is used for inflection generation.

3 Formal Task Description

Given a corpus $\mathcal{D} = w_1, \ldots, w_{|\mathcal{D}|}$ with a vocabulary V of word types $\{w_i\}$ and a lexicon $\mathcal{L} = \{\ell_j\}$ with $|\mathcal{L}|$ lemmas belonging to the same part of speech, the task of unsupervised morphological paradigm completion consists of generating the paradigms $\{\pi(\ell)\}_{\ell \in \mathcal{L}}$ of the entries in the lexicon.

Following Matthews and Matthews (1972) and Aronoff (1976), we treat a paradigm as a *vector of inflected forms* belonging to a lemma ℓ. Paradigm completion consists of predicting missing slots in the paradigm $\pi(\ell)$:

$$\pi(\ell) = \left\langle f(\ell, \vec{t}_\gamma) \right\rangle_{\gamma \in \Gamma(\ell)}, \qquad (1)$$

where $f : \Sigma^* \times \mathcal{T} \to \Sigma^*$ transforms a lemma into an inflected form,[4] $\vec{t}_\gamma \in \mathcal{T}$ is a vector of inflectional features describing paradigm slot γ, and $\Gamma(\ell)$ is the set of slots in lemma ℓ's paradigm. Since we only consider lemmas that belong to the same part of speech, we will use Γ and $\Gamma(\ell)$ interchangeably in the following. Furthermore, we will denote $f(\ell, \vec{t}_\gamma)$ as $f_\gamma(\ell)$ for simplicity.

Remarks on Task Design In general, not all paradigm entries will be present in the corpus $\mathcal{D}$. Thus, the task requires more than a keyword search.

On another note, it is not necessary to predict the features $\vec{t}_\gamma$ corresponding to each slot γ; as the exact denotation of features is up to human annotators, they cannot be inferred by unsupervised systems. For our task, it is enough to predict the ordered vector $\pi(\ell)$ of inflected forms.

4 Methodology

Our system implicitly solves two subtasks: (i) determining the number of paradigm slots; and (ii) generating the inflected form corresponding to each paradigm slot for each lemma. It is organized as a
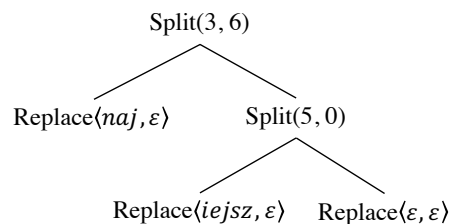

Figure 2: Visualization of the EDIT TREE constructed from *najtrudniejszy* to *trudny* (Chrupała, 2008).

pipeline consisting of multiple components. Our system is highly modular: individual components can be exchanged easily. In the remainder of this section, we will dedicate each subsection to one component.

4.1 Retrieval of Relevant EDIT TREES

The first component in our pipeline identifies words in the corpus $\mathcal{D}$ which could belong to the paradigm of one of the lemmas in the lexicon $\mathcal{L}$. We call those words *paradigm candidates*. It then uses the discovered paradigm candidates to identify EDIT TREE (Chrupała, 2008) operations that correspond to valid inflections.

Paradigm Candidate Discovery For most paradigms, all participating inflected forms share some characteristics—usually substrings—which humans use to identify the paradigm any given word belongs to. Given a pair (ℓ, w) of a lemma ℓ and a word form w, the first step in our pipeline is to determine whether w is a paradigm candidate for lemma ℓ. For example, *studied* is likely to be an inflected form of the English lemma *study*, while *monkey* is not. We identify paradigm candidates C_ℓ of a lemma ℓ by computing the longest common substring (LCS) between ℓ and w for all words w in the vocabulary V. If the ratio between the LCS's length and the length of ℓ is higher than a threshold λ_P, w is a paradigm candidate for ℓ:

$$C_\ell = \left\{ w \in V \left| \frac{|\text{LCS}(\ell, w)|}{|\ell|} > \lambda_P \right. \right\}. \qquad (2)$$

EDIT TREE Discovery Surface form changes, which we denote as ψ, define a modification of a word's surface form. Our system employs EDIT TREES (Chrupała, 2008) to represent ψ.

Given two strings $x_{[1\ldots n]}$ and $x'_{[1\ldots m]}$, EDIT TREE$(x_{[1\ldots n]}, x'_{[1\ldots m]})$ is constructed by first determining the LCS between x and x'. We then recursively model the substrings before and after the LCS. If the length of the LCS is zero, the

6698

EDIT TREE consists of the substitution operation of the first string with the second. For example, EDIT TREE(*najtrudniejszy*, *trudny*)[5] could be visualized as in Figure 2, where $\mathrm{Split}(i, j)$ represents taking the substring $x_{[i...n-j]}$.

An EDIT TREE can be applied to new input strings. The EDIT TREE in Figure 2, for example, could be applied to *najappleiejszs*, and the resulting string would be **apples**. Note that not all EDIT TREES can be applied to all strings. For example, the EDIT TREE in Figure 2 can only be applied to words starting with *naj*.

Our system constructs EDIT TREES from all pairs (ℓ, w) of lemmas ℓ and their paradigm candidates w and counts their frequencies:

$$n_\psi = \sum_{\substack{\ell \in \mathcal{L} \\ w \in C_\ell}} \mathbb{1} \left[\text{EDIT TREE}(\ell, w) = \psi \right]. \quad (3)$$

It then discards EDIT TREES with frequencies n_ψ below a threshold $\lambda_{FC}(|\mathcal{L}|)$, which is a function of the size of the lexicon $\mathcal{L}$:

$$\lambda_{FC}(|\mathcal{L}|) = \max \left\{ 2, \phi_{FC} \cdot |\mathcal{L}| \right\}, \quad (4)$$

where $\phi_{FC} \in \mathbb{R}$ is a hyperparameter. The idea is that an EDIT TREE is only valid if it can be applied to multiple given lemmas. EDIT TREES which we observe only once are always considered unreliable. Our system then retains a set of frequent surface form changes

$$\Psi = \{ \psi \mid n_\psi \geq \lambda_{FC}(|\mathcal{L}|) \} \quad (5)$$

represented by EDIT TREES. Assuming an one-to-one mapping between surface form changes and paradigm slots (that is, that $|\Psi| = |\Gamma|$ and that each ψ is equivalent to a particular inflection function f_γ), we now have a first basic paradigm completion system (**PCS-I**), which operates by applying all suitable EDIT TREES to all lemmas in our lexicon.

Complexity and Runtime The time complexity to compute EDIT TREE$(x_{[1...n]}, x'_{[1...m]})$ is $O(\max^3\{n, m\})$. Computing EDIT TREES can trivially be parallelized, and in practice this computation does not take much time.

4.2 Retrieval of Additional Lemmas

Since we assume a low-resource setting, our lexicon is small (≤ 100 entries). However, the more

---

[5]Polish form–lemma pair meaning *hardest* and *hard*. Example from Chrupała (2008).

lemmas we have, the more confident we can be that the EDIT TREES retrieved by the first component of our system represent valid inflections. An intuitive method to obtain additional lemmas would be to train a lemmatizer and to generate new lemmas from words in our corpus. However, due to the limited size of our initial lemma list, such a lemmatizer would most likely not be reliable.

The second component of our system employs another method, which guarantees that additionally retrieved lemmas are valid words: It is based on the intuition that a word $w \in V$ is likely to be a lemma if the pseudo–inflected forms of w, obtained by applying the EDIT TREES from §4.1, also appear in V. For a word $w \in V$, we say it is a discovered lemma if $w \notin \mathcal{L}$ and

$$\sum_{\psi \in \Psi} \mathbb{1} \left[\psi(w) \in V \right] > \lambda_{NL}(|\Psi|) \quad (6)$$

for

$$\lambda_{NL}(|\Psi|) = \max \left\{ 3, \phi_{NL} \cdot |\Psi| \right\}, \quad (7)$$

with $\phi_{NL} \in \mathbb{R}$ being a hyperparameter. Similar to Equation 4, λ_{NL} depends on the number of discovered EDIT TREES, but is never smaller than 3. We set this minimum to require evidence for at least two transformations in addition to the identity.

We can now bootstrap by iteratively computing additional paradigm candidates and EDIT TREES, and then retrieving more lemmas. We denote the paradigm completion systems resulting from one and two such iterations as **PCS-II-A** and **PCS-II-B**, respectively. Since we cannot be fully confident about retrieved lemmas, we associate each additional lemma with a weight $\theta_\ell = \theta_{NL}^{it}$, where θ_{NL} is a preset hyperparameter and it identifies the iteration in which a lemma is added, i.e., $it = 0$ for gold lemmas and $it = i$ for lemmas retrieved in the ith iteration. The weights θ_ℓ are used in later components of our system.

4.3 Paradigm Size Discovery

Until now, we have assumed a one-to-one mapping between paradigm slots and surface form changes. However, different EDIT TREES may indeed represent the same inflection. For example, the past tense inflection of verbs in English involves multiple EDIT TREES, as shown in Figure 3.

Thus, the next step is to group surface form changes based on the paradigm slots they realize.
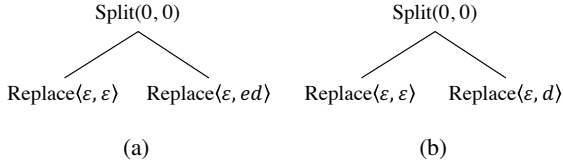
Figure 3: Visualization of the EDIT TREES representing (a) *work ↦ worked* and (b) *continue ↦ continued*.

4.3.1 One EDIT TREE per Lemma and Slot

Our algorithm for grouping surface form changes is based on two assumptions. First, since EDIT TREES are extracted from (ℓ, w) pairs, different EDIT TREES belonging to the same paradigm slot cannot be extracted from the same lemma.[6] Thus:

Assumption 1 *For each lemma, at most one inflected form per paradigm slot can be found in the corpus.*

Formally, for a multi-to-one mapping from EDIT TREES to paradigm slots $z : \Psi \to \Gamma$, we define the EDIT TREE set Ψ_γ of a potential paradigm slot γ as $\Psi_\gamma = \{\psi \mid z(\psi) = \gamma\}$, with $\bigcup_\gamma \Psi_\gamma = \Psi$. Then, for any lemma $\ell \in \mathcal{L}$ and proposed paradigm slot $\gamma \in \Gamma$, we have:

$$|\{w \in V \mid \psi(\ell) = w \wedge \psi \in \Psi_\gamma\}| \leq 1. \quad (8)$$

4.3.2 One Paradigm Slot per EDIT TREE

Our second assumption is a simplification,[7] but helps to reduce the search space during clustering:

Assumption 2 *Each surface form change $\psi \in \Psi$ belongs to exactly one paradigm slot.*

Formally, we partition Ψ into disjoint subsets:

$$\Psi_\gamma \cap \Psi_{\gamma'} = \emptyset \qquad \forall \gamma \neq \gamma'. \quad (9)$$

4.3.3 Paradigm Slot Features and Similarity

In addition to Assumptions 1 and 2, we make use of a feature function $r(\gamma)$ and a score function $s(r(\gamma), r(\gamma'))$, which measures the similarity between two potential paradigm slots.

Our *feature function* makes the connection between paradigm slots and the instances of inflected forms in the corpus by utilizing the part-of-speech (POS) tags as context information. In our implementation, we employ an unsupervised POS tagger

---

[6] Exceptions to this do exist. However, they are rare enough that we do not expect them to hurt our algorithm.

[7] This assumption ignores syncretism.

[8] Note that this example is simplified. The system does not need to actually know the tags like PST and V.

Figure 4: An example of the distributionally informed feature function with window size 3 for the past tense slot (PST). $stop \in \mathcal{L}$ and $f_{\text{PST}}(stop) = stopped$. When the sliding window arrives to this instance of *stopped*, $\vec{r}^{\,\text{PST}}_{[\text{N},\text{V},\text{V}]}$ is increased by 1.[8]

(Stratos et al., 2016) to extract tags for each word in the corpus $\mathcal{D}$.

This tagger assigns an *anchor word*, i.e., a pseudo POS tag, to each word w_i in the corpus by using an **anchor hidden Markov model (HMM)** with 8 hidden states. Our feature function counts the tag tuples within a sliding window centered on each instance of inflected forms of a potential paradigm slot.

Formally, we denote the set of lemmas that surface form change ψ is applied to as $\mathcal{L}_\psi$, the set of lemmas that express a potential paradigm slot γ as $\mathcal{L}_\gamma$, and the corresponding inflected forms as V_γ:

$$\mathcal{L}_\psi = \{\ell \in \mathcal{L} \mid \psi(\ell) = w \wedge w \in V\} \quad (10)$$

$$\mathcal{L}_\gamma = \bigcup_{\psi \in \Psi_\gamma} \mathcal{L}_\psi \quad (11)$$

$$V_\gamma = \bigcup_{\psi \in \Psi_\gamma} \{\psi(\ell)\}_{\ell \in \mathcal{L}_\psi}. \quad (12)$$

We further refer to the set of available POS tag labels as $P = \{p_1, \ldots, p_8\}$. For a corpus $\mathcal{D} = w_1, \ldots, w_{|\mathcal{D}|}$, a window size $2d+1$, and a potential paradigm slot γ, our feature function is defined as:

$$\vec{r}^{\,\gamma} = r(\gamma; \mathcal{D}, 2d+1), \quad (13)$$

where $\vec{r}^{\,\gamma}$ is a vector with one dimension for each possible tag tuple sequence of length 2d+1. Its values are computed as:

$$\vec{r}^{\,\gamma}_{\left[p_{j_1}, \ldots, p_{j_{2d+1}}\right]} = \sum_{w_i \in \mathcal{D}} \mathbb{1}\big[w_i \in V_\gamma \quad (14)$$
$$\wedge\, \hat{p}_{i-d} = p_{j_1} \wedge \cdots \wedge \hat{p}_{i+d} = p_{j_{2d+1}} \big]$$
$$\forall \left(\hat{p}_{j_1}, \ldots, \hat{p}_{j_{2d+1}} \right) \in P^{2d+1}.$$

In practice, we initialize $\vec{r}^{\,\gamma}$ to the zero vector, and iterate the sliding window over the corpus. When the central word $w_i \in V_\gamma$, the corresponding value of $\vec{r}^{\,\gamma}$ at the POS tuple within the sliding window is incremented by 1. Figure 4 shows an example in English.

We assume that paradigm slots γ and γ' are similar if the words in V_γ and $V_{\gamma'}$ frequently appear in

Algorithm 1: Surface Form Change Grouping

Result: $\Gamma, \{\Psi_\gamma\}_{\gamma \in \Gamma}$
Initialize Γ s.t. $|\Gamma| = |\Psi|$ and $f_{\gamma_i} = \psi_i$ for all $\psi_i \in \Psi$;
Initialize $\Psi_{\gamma_i} = \{\psi_i\}$ for all $\psi_i \in \Psi$;
while $\exists (\gamma, \gamma')$ s.t. $\mathcal{L}_\gamma \cap \mathcal{L}_{\gamma'} = \emptyset$ **do**
  $\overrightarrow{score} \leftarrow \vec{0}$;
  for (γ, γ') s.t. $\mathcal{L}_\gamma \cap \mathcal{L}_{\gamma'} = \emptyset$ **do**
    $\overrightarrow{score}_{(\gamma,\gamma')} \leftarrow s(r(\gamma), r(\gamma'))$;
  end
  $(\hat{\gamma}, \hat{\gamma}') = \arg\max_{(\gamma,\gamma')} \{\overrightarrow{score}_{(\gamma,\gamma')}\}$;
  if $\overrightarrow{score}_{(\hat{\gamma}, \hat{\gamma}')} > \lambda_S$ **then**
    denote the new paradigm slot γ_{merge};
    $\Psi_{\gamma_{\text{merge}}} = \Psi_{\hat{\gamma}} \cup \Psi_{\hat{\gamma}'}$;
    $\Gamma \leftarrow (\Gamma \setminus \{\hat{\gamma}, \hat{\gamma}'\}) \cup \{\gamma_{\text{merge}}\}$;
  else
    break;
  end
end

---

similar contexts, i.e., within similar tag sequences. With the feature function defined above, our system uses *cosine similarity* as the score function s.

We then develop Algorithm 1 to group one-to-one mappings from surface form changes to paradigm slots into many-to-one mappings. The idea is to iteratively merge the most similar slots if this is not violating Assumption 1 until the similarity gets too low. $\lambda_S \in (0, 1)$ is a threshold parameter.[9]

4.4 Generation

Now, one paradigm slot can be represented by multiple EDIT TREES. Our system, thus, needs to learn to apply the correct transformation for a combination of lemma and paradigm slot. However, mapping lemmas and paradigm slots to inflected forms corresponds exactly to the morphological inflection task, which has been the subject of multiple shared tasks over the last years (Cotterell et al., 2018).

Our morphological inflection models take (slot, lemma, word) tuples extracted by the previous components of our system as training data. Formally, they are trained on the training set:

$$\left\{ (\vec{t}_\gamma, \ell, f_\gamma(\ell)) \mid \gamma \in \Gamma \wedge \ell \in \mathcal{L}_\gamma \wedge f_\gamma(\ell) \in V \right\}. \tag{15}$$

We explore two morphological inflection models from the literature.

Affix Editing The baseline system of the CoNLL–SIGMORPHON 2017 shared task[10] (Cotterell et al., 2017) is a simple approach, which is very suitable for low-resource settings. The system breaks each word into PREFIX, STEM, and SUFFIX, and then stores the PREFIX editing rules and the STEM+SUFFIX editing rules. At test time, it applies the longest possible PREFIX and STEM+SUFFIX editing rules to the input lemma. We denote the surface form change grouping in combination with this system as **PCS-III-C**.

Transducer-Based Hard-Attention We further experiment with a transducer-based hard-attention model (Makarov and Clematide, 2018a). Unlike widely used soft-attention sequence-to-sequence models (Bahdanau et al., 2015), which predict the target tokens directly, it predicts edit action sequences to transform the input sequence into outputs, and it disposes of a hard attention mechanism. We denote the surface form change grouping in combination with this system as **PCS-III-H**.

5 Experiments

5.1 Data

To evaluate our approach in a real-world setting, we restrict our data to resources typically available to a field linguist: a small written corpus ($\leq$ 100k tokens) and a small lexicon.

For our corpora, we use the JHU Bible Corpus (McCarthy et al., 2020), which allows future work to build systems in 1600 languages. The Bible is frequently available even in low-resource languages: Ethnologue identifies 3,995 written languages, and the New Testament has been translated into 2,246. The Bible is also highly representative of a language's *core vocabulary*: Resnik et al. (1999) find high overlap with both the Longman Dictionary of Contemporary English (Summers and Gadsby, 1995) and the Brown Corpus (Francis and Kucera, 1964). Furthermore, the Bible is multi-parallel and, thus, allows for a fair comparison across languages without confounds like domain.

For *evaluation* of our methods only, we additionally obtain ground-truth morphological paradigms from UniMorph (Kirov et al., 2018), which provides paradigms for over 100 languages.

From the intersection of languages in the Bible and UniMorph, we select 14 typologically diverse

---

[9]This does not result in a functional paradigm completion system, since we still lack a method to decide which surface form change to apply to realize a paradigm slot for a lemma.

[10]https://github.com/sigmorphon/conll2017/tree/master/baseline

languages from 7 families, each of which display inflectional morphology: Basque (EUS), Bulgarian (BUL), English (ENG), Finnish (FIN), German (DEU), Kannada (KAN), Navajo (NAV), Spanish (SPA), and Turkish (TUR) as test languages, and Maltese (MLT), Persian (FAS), Portuguese (POR), Russian (RUS), and Swedish (SWE) for development. To create test data for all and development data for our development languages, we sample 100 paradigms for each set from UniMorph, then take their lemmas as our lexicon $\mathcal{L}$.[11]

5.2 Baselines and Skylines

Lemma Baseline (LB)   Our first, trivial baseline predicts inflected forms identical to the lemma for all paradigm slots. We compare to one version of this baseline that has access to the ground-truth paradigm size (**LB-Truth**), and a second version which predicts the paradigm size as the average over the development languages (**LB-Dev**).

One/Ten-Shot Inflection Model   Our second baseline could be seen as a *skyline*, since it leverages morphological information our proposed system does not have access to. In particular, we train the baseline system of CoNLL–SIGMORPHON 2017 (Cotterell et al., 2017) on one (**CoNLL17-1**) and ten (**CoNLL17-10**) paradigms. For this, we randomly sample paradigms from UniMorph, excluding those in our test data.

5.3 Hyperparameters

We choose the hyperparameters by grid search over intuitively reasonable ranges, using the development languages. No test language data is seen before final testing. Note also that only the corpus and the lexicon can be accessed by our system, and no ground-truth morphological information (including paradigm size) is given.

Our final hyperparameters are $\lambda_P = 0.5$, $\phi_{FC} = 0.05$, $\phi_{NL} = 0.2$, $\theta_{NL} = 0.5$, and $\lambda_S = 0.3$. The window size $2d+1$ for feature extraction is set to 3. For **CoNLL17-1**, we average over the results for 10 different sampled paradigms for each language.

5.4 Evaluation Metrics

Systems for *supervised* or *semi-supervised* paradigm completion are commonly being evaluated using word-level accuracy (Dreyer and Eisner, 2011; Cotterell et al., 2017). However, this is not possible for our task because our system cannot access the gold data paradigm slot descriptions and, thus, does not necessarily produce one word for each ground-truth inflected form. Furthermore, the system outputs pseudo-tags, and the mapping from pseudo-tags to paradigm slots is unknown.

Therefore, we propose to use *best-match accuracy* (BMAcc), the best accuracy among all mappings from pseudo-tags to paradigm slots, for evaluation. Let $\Gamma = \{\gamma_i\}_{i=1}^{N}$ and $\hat{\Gamma} = \{\hat{\gamma}_j\}_{j=1}^{M}$ be the set of all paradigm slots in the ground truth and the prediction, respectively, with transformation functions $f : \Sigma^* \times \Gamma \to \Sigma^* \cup \{\emptyset\}$ and $\hat{f} : \Sigma^* \times \hat{\Gamma} \to \Sigma^*$,[12] where $f_\gamma(\ell) = \emptyset$ if the corresponding inflection is missing in the ground truth.[13] We define two types of BMAcc:

Macro-averaged BMAcc   This is the average *per-slot accuracy* for the best possible matching of slots. For any $\gamma_i, \hat{\gamma}_j$, we define $g_t(\mathcal{L}, \gamma_i, \hat{\gamma}_j)$ as the number of correct guesses (true positives) if $\hat{\gamma}_j$ maps to γ_i, $g_a(\mathcal{L}, \gamma_i)$ as the number of ground truth inflections for γ_i, and $\mathrm{acc}(\mathcal{L}, \gamma_i, \hat{\gamma}_j)$ as the per-slot accuracy:

$$g_t(\mathcal{L}, \gamma_i, \hat{\gamma}_j) = |\{\ell \in \mathcal{L} \mid f_{\gamma_i}(\ell) = \hat{f}_{\gamma_j}(\ell) \neq \emptyset\}|$$
$$g_a(\mathcal{L}, \gamma_i) = |\{\ell \in \mathcal{L} \mid f_{\gamma_i}(\ell) \neq \emptyset\}|, \qquad (16)$$

and

$$\mathrm{acc}(\mathcal{L}, \gamma_i, \hat{\gamma}_j) = \frac{g_t(\mathcal{L}, \gamma_i, \hat{\gamma}_j)}{g_a(\mathcal{L}, \gamma_i)}. \qquad (17)$$

Then, we construct a complete bipartite graph with Γ and $\hat{\Gamma}$ as two sets of vertices and $\mathrm{acc}(\mathcal{L}, \gamma_i, \hat{\gamma}_j)$ as edge weights. The maximum-weight full matching can be computed efficiently with the algorithm of Karp (1980). With such a matching $\mathcal{M} = \{(\gamma_m, \hat{\gamma}_m)\}_{m=0}^{\min\{N,M\}}$, the macro-averaged BMAcc is defined as:

$$\mathrm{BMAcc\text{-}macro}(\mathcal{L}, \Gamma, \hat{\Gamma}) \qquad (18)$$
$$= \frac{\sum_{(\gamma_m, \hat{\gamma}_m) \in \mathcal{M}} \mathrm{acc}(\mathcal{L}, \gamma_m, \hat{\gamma}_m)}{\max\{N, M\}}$$

The normalizing factor $\frac{1}{\max\{N,M\}}$ rewards predicting the correct number of paradigm slots. In the case when $\mathrm{acc}(\mathcal{L}, \gamma_m, \hat{\gamma}_m) = 1$ for all $(\gamma_m, \hat{\gamma}_m) \in \mathcal{M}$, BMAcc-macro$(\mathcal{L}, \Gamma, \hat{\Gamma})$ reaches its maximum if and only if $N = M$.

---

[11]For Basque, Kannada, and Maltese, we only take 20 paradigms for each set, due to limited availability.

[12]These are equivalent to our definition in §3, since the system does not need to predict the inflectional features.

[13]In practice, we merge paradigm slots that are identical for all lemmas in both the predictions and gold standard before evaluating.

Method	Test Ave.	EUS	BUL	ENG	FIN	DEU	KAN	NAV
				Test				
LB-Truth	4.94 / 5.25	0.03 / 0.05 (1659)	1.89 / 2.16 (56)	20.40 / 20.40 (5)	0.96 / 0.96 (141)	15.66 / 15.68 (29)	1.18 / 1.19 (85)	**2.89 / 5.39** (50)
LB-Dev	2.22 / 2.53	0.03 / 0.05 (48)	1.89 / 2.16 (48)	2.13 / 2.15 (48)	0.96 / 0.96 (48)	9.46 / 9.47 (48)	1.18 / 1.19 (48)	**2.89 / 5.39** (48)
CoNLL17-1	18.70 / 18.70	0.00 / 0.00 (1659)	12.34 / 12.24 (56)	59.92 / 59.92 (5)	3.52 / 3.52 (141)	26.71 / 26.73 (29)	5.74 / 5.82 (85)	0.00 / 0.00 (50)
CoNLL17-10	**35.58 / 35.56**	0.00 / 0.00 (1659)	**43.75 / 43.58** (56)	70.58 / 70.58 (5)	**24.51 / 24.51** (141)	**35.75 / 35.77** (29)	**34.53 / 34.51** (85)	0.00 / 0.00 (50)
PCS-I	16.09 / 16.39	<u>0.11</u> / <u>0.11</u> (51)	18.01 / 19.18 (39)	56.17 / 56.17 (6)	3.40 / 3.40 (23)	18.28 / 18.30 (19)	14.29 / 14.15 (209)	2.19 / 3.74 (8)
PCS-I+II-A	17.60 / 17.89	<u>0.11</u> / <u>0.11</u> (51)	<u>22.72</u> / <u>23.87</u> (52)	56.17 / 56.17 (6)	<u>5.70</u> / <u>5.70</u> (71)	20.57 / 20.56 (32)	12.60 / 12.48 (237)	2.19 / 3.74 (19)
PCS-I+II-B	17.60 / 17.89	<u>0.11</u> / <u>0.11</u> (51)	<u>22.72</u> / <u>23.87</u> (57)	56.17 / 56.17 (6)	<u>5.70</u> / <u>5.70</u> (71)	20.57 / 20.56 (32)	12.60 / 12.48 (237)	2.19 / 3.74 (18)
PCS-I+III-C	15.90 / 16.19	0.07 / 0.10 (27)	16.18 / 17.24 (16)	63.20 / 63.20 (5)	3.58 / 3.58 (15)	5.19 / 5.18 (14)	17.82 / 17.82 (162)	2.19 / 3.74 (8)
PCS-I+III-H	17.45 / 17.67	0.04 / 0.04 (27)	16.95 / 18.16 (16)	62.20 / 62.20 (5)	3.78 / 3.78 (15)	19.98 / 19.96 (14)	15.83 / 15.85 (162)	<u>2.55</u> / <u>4.78</u> (8)
PCS-I+II+III-C	<u>19.10</u> / <u>19.41</u>	0.07 / 0.10 (27)	16.77 / 17.94 (16)	72.80 / 72.80 (4)	3.51 / 3.51 (15)	<u>23.28</u> / <u>23.31</u> (13)	<u>18.03</u> / <u>17.90</u> (161)	2.16 / 3.81 (7)
PCS-I+II+III-H	18.76 / 19.06	0.06 / 0.08 (27)	17.39 / 18.65 (16)	**74.00 / 74.00** (4)	3.77 / 3.77 (15)	18.87 / 18.89 (13)	17.90 / <u>17.90</u> (161)	2.0 / 3.41 (7)

Method	SPA	TUR	Dev. Ave.	MLT	FAS	POR	RUS	SWE
	Test		**Development**					
LB-Truth	1.43 / 1.43 (70)	0.00 / 0.00 (120)	7.94 / 7.87	9.21 / 7.69 (24)	2.04 / 2.04 (136)	6.53 / 6.53 (76)	5.47 / 5.17 (55)	16.44 / 17.91 (11)
LB-Dev	1.43 / 1.43 (48)	0.00 / 0.00 (48)	4.33 / 4.18	4.25 / 3.55 (52)	2.04 / 2.04 (33)	6.53 / 6.53 (43)	5.47 / 5.17 (47)	3.35 / 3.65 (54)
CoNLL17-1	60.07 / 60.07 (70)	0.00 / 0.00 (120)	33.68 / 34.73	11.82 / 12.08 (24)	46.68 / 46.68 (136)	82.87 / 82.87 (76)	7.75 / 10.98 (55)	19.30 / 21.03 (11)
CoNLL17-10	**76.23 / 76.23** (70)	**34.88 / 34.88** (120)	49.3 / 53.02	13.07 / **13.33** (24)	**64.26 / 64.26** (136)	**89.54 / 89.54** (76)	**19.22 / 36.75** (55)	**60.41 / 61.22** (11)
PCS-I	25.04 / 25.04 (30)	7.40 / 7.40 (186)	18.72 / 21.66	12.21 / <u>12.00</u> (25)	6.36 / 6.36 (35)	36.47 / 36.47 (45)	8.24 / 20.98 (22)	30.31 / 32.47 (20)
PCS-I+II-A	<u>31.03</u> / <u>31.03</u> (39)	7.33 / 7.33 (237)	17.06 / 20.77	8.45 / 8.50 (38)	6.50 / 6.50 (52)	<u>37.14</u> / <u>37.14</u> (92)	12.56 / 30.66 (53)	20.65 / 21.06 (33)
PCS-I+II-B	<u>31.03</u> / <u>31.03</u> (39)	7.33 / 7.33 (237)	17.02 / 20.80	7.83 / 7.88 (41)	<u>6.51</u> / <u>6.51</u> (52)	<u>37.14</u> / <u>37.14</u> (92)	12.95 / <u>31.42</u> (55)	20.65 / 21.06 (33)
PCS-I+III-C	24.10 / 24.10 (29)	10.73 / 10.73 (91)	22.43 / 23.78	<u>15.79</u> / 11.54 (17)	6.24 / 6.24 (30)	27.22 / 27.22 (29)	11.56 / 20.48 (15)	51.36 / 53.43 (11)
PCS-I+III-H	24.36 / 24.36 (29)	<u>11.69</u> / <u>11.69</u> (91)	21.27 / 23.13	14.69 / 9.94 (17)	4.90 / 4.90 (30)	28.36 / 28.36 (29)	7.98 / 20.12 (15)	50.40 / 52.34 (11)
PCS-I+II+III-C	24.01 / 24.01 (29)	11.35 / 11.35 (91)	22.74 / 24.79	8.55 / 6.73 (13)	6.34 / 6.34 (29)	31.71 / 31.71 (31)	12.02 / 21.74 (15)	<u>55.07</u> / <u>57.41</u> (11)
PCS-I+II+III-H	24.30 / 24.30 (29)	10.54 / 10.54 (91)	<u>23.53</u> / <u>25.43</u>	12.94 / 11.22 (13)	6.35 / 6.35 (29)	32.43 / 32.43 (31)	<u>13.37</u> / 22.05 (15)	52.58 / 55.12 (11)

Table 1: Macro- and micro-averaged BMAcc (in percentage) as well as the predicted number of paradigm slots (in brackets), for each method. Overall best scores are **bold**, and the best scores of our system are <u>underlined</u>.

Micro-averaged BMAcc Our second metric is conceptually closer to word-level accuracy. We start with the same process of bipartite graph matching, but instead use $g_t(\mathcal{L}, \gamma_i, \hat{\gamma}_j)$ as edge weights. Given the optimal matching $\mathcal{M}$, the micro-averaged BMAcc is defined as:

$$\text{BMAcc-micro}(\mathcal{L}, \Gamma, \hat{\Gamma}) = \frac{N}{\max\{N, M\}} \frac{\sum_{(\gamma_m, \hat{\gamma}_m) \in \mathcal{M}} g_t(\mathcal{L}, \gamma_m, \hat{\gamma}_m)}{\sum_{\gamma_i \in \Gamma} g_a(\mathcal{L}, \gamma_i)}. \quad (19)$$

5.5 Results and Discussion

Overall Results We present our experimental results in Table 1. The performance of our system varies widely across languages, with best results for ENG (74% BMAcc). On average over languages, our final system obtains 18.76%/19.06% BMAcc on the test set, as compared to the baseline of 4.94%/5.25% and skylines of 18.70%/18.70% and 35.58%/35.56%. Compared to versions of our system without selected components, our final system performs best on average for both development and test languages. Leaving out step II or step III leads to a reduction in performance.

Notably, variants of our system outperform the *skyline* CoNLL17-1, which has seen one training example and, thus, knows the correct paradigm size in advance, on EUS, BUL, ENG, FIN, KAN, NAV, TUR, MLT, RUS, and SWE. Moreover, it even outperforms CoNLL17-10 on EUS, ENG, and NAV,

which shows that unsupervised paradigm completion has promise even in cases where a limited number of training examples—but not large amounts—are available.

Differences between Languages We hypothesize that the large differences between languages—over 73% between EUS and ENG—can be explained in parts by the following reasons:

Intuitively, the larger the paradigm, the more difficult the task. If the number of slots is huge, each individual inflection is rare, and it is hard for any unsupervised paradigm completion system to distinguish true inflections (e.g., *rise → rises*) from false candidates (e.g., *rise → arise*). This could explain the high performance on ENG and the low performance on EUS, FIN, and FAS.

Related to the last point, in a limited corpus such as the Bible, some inflected forms might not appear for any lemma, which makes them undetectable for unsupervised paradigm completion systems. For example, a FAS paradigm has 136 slots in Uni-Morph, but only 46 are observed.[14] Additional statistics can be found in Table 2.

Furthermore, Assumption 2 does not hold for all languages. Surface forms can be shared between paradigm slots, as, for instance, in English for *he studied* and *he has studied*. Different languages

---

[14] We had to estimate that number based on available data in UniMorph: a slot is considered observed if at least one inflected form for that slot can be found.

Language	ET Match	Rep. Words	Absent/Total Slots
MLT	18.13	58.18	0/16
FAS	12.10	28.66	90/136
POR	56.52	30.56	4/76
RUS	24.69	25.45	4/55
SWE	72.91	13.98	0/11

Table 2: Statistics for our development languages, computed with UniMorph. *ET Match* is the percentage of gold (ℓ, w) pairs that can be matched to an EDIT TREE. *Rep. Words* denotes the percentage of inflected forms that represent multiple paradigm slots. *Absent/Total Slots* is the numbers of unobservable and total slots.

show different degrees of this phenomenon called *syncretism*.

Pipeline Effects Different combinations of components result in major performance differences. In particular, each step of our system has the potential to introduce errors. This demonstrates a pitfall of pipeline methods also discussed in McCarthy et al. (2020): the quality of individual steps, here, e.g., EDIT TREE discovery and retrieval of additional lemmas, can greatly affect the results of PCS-II and PCS-III.

Differences in Components Details of individual components also affect the results. On the one hand, applying more than one iteration of additional lemma retrieval impacts the results only slightly, as those lemmas are assigned very small weights. On the other hand, we see performance differences $> 2\%$ between PCS-III-C and PCS-III-H for DEU, MLT, and SWE.

Analysis of EDIT TREE Quality As it is the first step in our pipeline, the quality of the EDIT TREE discovery strongly affects the performance of later components. For our development languages, we show in Table 2 the percentage of (ℓ, w) pairs for which the system predicts an EDIT TREE ψ such that $\psi(\ell) = w$ appears in the gold paradigm of ℓ. This corresponds to the highest possible performance after PCS-I. FAS has the worst performance (12.10%), while the results for SWE are high (72.91%). As expected, languages with lower values here also obtain lower final results.

Analysis of Syncretism We further hypothesize that syncretism could be a source of errors, due to Assumption 2. Table 2 shows the percentage of words that are the inflected forms corresponding to multiple paradigm slots of the same lemma.

We observe that SWE has a low degree of syncretism, and, in fact, our system predicts the correct paradigm size for SWE. A high degree of syncretism, in contrast, might contribute to the low performance on MLT.

6 Conclusion

We proposed *unsupervised morphological paradigm completion*, a novel morphological generation task. We further developed a system for the task, which performs the following steps: (i) EDIT TREE retrieval, (ii) additional lemma retrieval, (iii) paradigm size discovery, and (iv) inflection generation. Introducing best-match accuracy, a metric for the task, we evaluated our system on a typologically diverse set of 14 languages. Our system obtained promising results for most of our languages and even outperformed a minimally supervised baseline on Basque, English, and Navajo. Further analysis showed the importance of our individual components and detected possible sources of errors, like wrongly identified EDIT TREES early in the pipeline or syncretism.

In the future, we will explore the following directions: (i) A difficult challenge for our proposed system is to correctly determine the paradigm size. Since transfer across related languages has shown to be beneficial for morphological tasks (Jin and Kann, 2017; McCarthy et al., 2019; Anastasopoulos and Neubig, 2019, *inter alia*), future work could use typologically aware priors to guide the number of paradigm slots based on the relationships between languages. (ii) We plan to explore other methods, like word embeddings, to incorporate context information into our feature function. (iii) We aim at developing better performing string transduction models for the morphological inflection step. By substituting the current transducers in our pipeline, we expect that we will be able to improve the overall performance of our system.

References

Roee Aharoni and Yoav Goldberg. 2017. Morphological inflection generation with hard monotonic attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2004–2015, Vancouver, Canada. 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, Denver, Colorado. Association for Computational Linguistics.

Antonios Anastasopoulos and Graham Neubig. 2019. Pushing the limits of low-resource morphological inflection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 984–996, Hong Kong, China. Association for Computational Linguistics.

Mark Aronoff. 1976. Word formation in generative grammar. In *Linguistic Inquiry Monographs*, 1, pages 1–134. MIT Press, Cambridge, Mass.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR*.

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

Jean Berko. 1958. The child's learning of English morphology. *Word*, 14(2-3):150–177.

Grzegorz Chrupała. 2008. *Towards a machine-learning architecture for lexical functional grammar parsing*. Ph.D. thesis, Dublin City University.

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, Brussels. Association for Computational Linguistics.

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, Vancouver. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. 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. Association for Computational Linguistics.

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

Mathias Creutz, Teemu Hirsimäki, Mikko Kurimo, Antti Puurula, Janne Pylkkönen, Vesa Siivola, Matti Varjokallio, Ebru Arisoy, Murat Saraçlar, and Andreas Stolcke. 2007. Analysis of morph-based speech recognition and the modeling of out-of-vocabulary words across languages. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 380–387, Rochester, New York. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(1):1–34.

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

Greg Durrett and John DeNero. 2013. Supervised learning of complete morphological paradigms. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1185–1195, Atlanta, Georgia. Association for Computational Linguistics.

Alexander Erdmann, Micha Elsner, Shijie Wu, Ryan Cotterell, and Nizar Habash. 2020. The paradigm discovery problem. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

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. Association for Computational Linguistics.

W Nelson Francis and Henry Kucera. 1964. Brown corpus. *Department of Linguistics, Brown University, Providence, Rhode Island*, 1.

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

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6097–6110, Hong Kong, China. Association for Computational Linguistics.

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

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

Huiming Jin and Katharina Kann. 2017. Exploring cross-lingual transfer of morphological knowledge in sequence-to-sequence models. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 70–75, Copenhagen, Denmark. Association for Computational Linguistics.

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2017. One-shot neural cross-lingual transfer for paradigm completion. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1993–2003, Vancouver, Canada. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2016. Single-model encoder-decoder with explicit morphological representation for reinflection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 555–560, Berlin, Germany. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2018. Neural transductive learning and beyond: Morphological generation in the minimal-resource setting. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3254–3264, Brussels, Belgium. Association for Computational Linguistics.

Richard M Karp. 1980. An algorithm to solve the $m \times n$ assignment problem in expected time $O(mn \log n)$. *Networks*, 10(2):143–152.

Aleksandr E. Kibrik. 1977. Opyt strukturnogo opisanija archinskogo jazyka. T. I. leksika. fonetika. T. II. taksonomicheskaja grammatika.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sebastian Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal morphology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Peter Makarov and Simon Clematide. 2018a. Imitation learning for neural morphological string transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2877–2882, Brussels, Belgium. Association for Computational Linguistics.

Peter Makarov and Simon Clematide. 2018b. Neural transition-based string transduction for limited-resource setting in morphology. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 83–93, Santa Fe, New Mexico, USA. 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, Vancouver. Association for Computational Linguistics.

Peter Hugoe Matthews and PH Matthews. 1972. *Inflectional morphology: A theoretical study based on aspects of Latin verb conjugation*, volume 6. CUP Archive.

A. D. McCarthy, L. Puzon, and J. Pino. 2020. SkinAugment: Auto-encoding speaker conversions for automatic speech translation. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7924–7928.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

6706

Arya D. McCarthy, Rachel Wicks, Dylan Lewis, Aaron Mueller, Winston Wu, Oliver Adams, Garrett Nicolai, Matt Post, and David Yarowsky. 2020. The Johns Hopkins University Bible Corpus: 1600+ tongues for typological exploration. In *Proceedings of the Twelfth International Conference on Language Resources and Evaluation (LREC 2020)*. European Language Resources Association (ELRA).

Karthik Narasimhan, Regina Barzilay, and Tommi Jaakkola. 2015. An unsupervised method for uncovering morphological chains. *Transactions of the Association for Computational Linguistics*, 3:157–167.

Karthik Narasimhan, Damianos Karakos, Richard Schwartz, Stavros Tsakalidis, and Regina Barzilay. 2014. Morphological segmentation for keyword spotting. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 880–885, Doha, Qatar. Association for Computational Linguistics.

Garrett Nicolai, Colin Cherry, and Grzegorz Kondrak. 2015. Inflection generation as discriminative string transduction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 922–931, Denver, Colorado. Association for Computational Linguistics.

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

Iwona Sadowska. 2012. *Polish: A comprehensive grammar*. Routledge.

Wolfgang Seeker and Özlem Çetinoğlu. 2015. A graph-based lattice dependency parser for joint morphological segmentation and syntactic analysis. *Transactions of the Association for Computational Linguistics*, 3:359–373.

Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL-08: HLT*, pages 737–745, Columbus, Ohio. Association for Computational Linguistics.

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, Denver, Colorado. Association for Computational Linguistics.

Karl Stratos, Michael Collins, and Daniel Hsu. 2016. Unsupervised part-of-speech tagging with anchor hidden Markov models. *Transactions of the Association for Computational Linguistics*, 4:245–257.

Della Summers and Adam Gadsby. 1995. *Longman dictionary of contemporary English*. Longman Harlow.

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 *Advances in neural information processing systems*, pages 5998–6008.

Hongzhi Xu, Mitchell Marcus, Charles Yang, and Lyle Ungar. 2018. Unsupervised morphology learning with statistical paradigms. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 44–54, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Chunting Zhou and Graham Neubig. 2017. Multi-space variational encoder-decoders for semi-supervised labeled sequence transduction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 310–320, Vancouver, Canada. Association for Computational Linguistics.