A Structured Variational Autoencoder for Contextual Morphological Inflection

Lawrence Wolf-Sonkin* **Jason Naradowsky*** **Sabrina J. Mielke*** **Ryan Cotterell***
Department of Computer Science, Johns Hopkins University
{lawrencews,narad,sjmielke,ryan.cotterell}@jhu.edu

Abstract

Statistical morphological inflectors are typically trained on fully supervised, type-level data. One remaining open research question is the following: How can we effectively exploit raw, token-level data to improve their performance? To this end, we introduce a novel generative latent-variable model for the semi-supervised learning of inflection generation. To enable posterior inference over the latent variables, we derive an efficient variational inference procedure based on the wake-sleep algorithm. We experiment on 23 languages, using the Universal Dependencies corpora in a simulated low-resource setting, and find improvements of over 10% absolute accuracy in some cases.

1 Introduction

The majority of the world's languages overtly encodes syntactic information on the word form itself, a phenomenon termed inflectional morphology (Dryer et al., 2005). In English, for example, the verbal lexeme with lemma talk has the four forms: talk, talks, talked and talking. Other languages, such as Archi (Kibrik, 1998), distinguish more than a thousand verbal forms. Despite the cornucopia of unique variants a single lexeme may mutate into, native speakers can flawlessly predict the correct variant that the lexeme's syntactic context dictates. Thus, in computational linguistics, a natural question is the following: Can we estimate a probability model that can do the same?

The topic of inflection generation has been the focus of a flurry of individual attention of late and, moreover, has been the subject of two shared tasks (Cotterell et al., 2016, 2017). Most work, however, has focused on the fully supervised case—a source lemma and the morpho-syntactic properties are fed into a model, which is asked to produce the desired inflection. In contrast, our work focuses on the semi-supervised case, where we wish to make use of unannotated raw text, i.e., a sequence of inflected tokens.

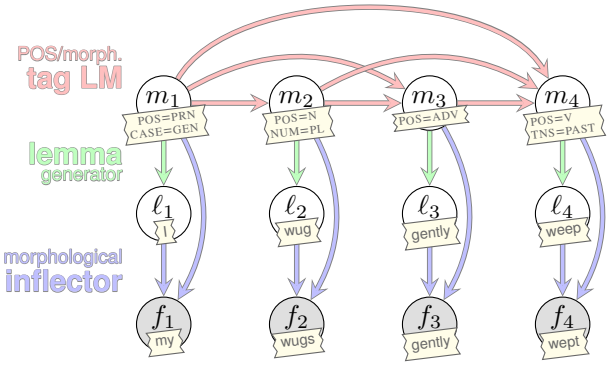


Figure 1: A length-4 example of our generative model factorized as in Eq. (1) and overlayed with example values of the random variables in the sequence. We highlight that all the conditionals in the Bayesian network are recurrent neural networks, e.g., we note that m_i depends on $\boldsymbol{m}_{<i}$ because we employ a recurrent neural network to model the morphological tag sequence.

Concretely, we develop a generative directed graphical model of inflected forms *in context*. A contextual inflection model works as follows: Rather than just generating the proper inflection for a single given word form *out of context* (for example *walking* as the gerund of *walk*), our generative model is actually a fully-fledged language model. In other words, it generates *sequences* of inflected words. The graphical model is displayed in Fig. 1 and examples of words it may generate are pasted on top of the graphical model notation. That our model is a language model enables it to exploit both inflected lexicons and unlabeled raw text in a prin-

* All authors contributed equally.

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 2631–2641
Melbourne, Australia, July 15 - 20, 2018.

	SG	PL	SG	PL
NOM	Wort	Wörter	Herr	Herren
GEN	Wortes	Wörter	Herrn	Herren
ACC	Wort	Wörter	Herrn	Herren
DAT	Worte	Wörtern	Herrn	Herren

Table 1: As an exhibit of morphological inflection, full paradigms (two numbers and four cases, 8 slots total) for the German nouns Wort ("word") and Herr ("gentleman"), with abbreviated and tabularized UniMorph annotation.

cipled semi-supervised way. In order to train using raw-text corpora (which is useful when we have less annotated data), we marginalize out the unobserved lemmata and morpho-syntactic annotation from unlabeled data. In terms of Fig. 1, this refers to marginalizing out $m_1, \ldots, m_4$ and $\ell_1, \ldots, \ell_4$. As this marginalization is intractable, we derive a variational inference procedure that allows for efficient approximate inference. Specifically, we modify the wake-sleep procedure of Hinton et al. (1995). It is the inclusion of raw text in this fashion that makes our model *token level*, a novelty in the camp of inflection generation, as much recent work in inflection generation (Dreyer et al., 2008; Durrett and DeNero, 2013; Nicolai et al., 2015; Ahlberg et al., 2015; Faruqui et al., 2016), trains a model on *type-level* lexicons.

We offer empirical validation of our model's utility with experiments on 23 languages from the Universal Dependencies corpus in a simulated low-resource setting.[1] Our semi-supervised scheme improves inflection generation by over 10% absolute accuracy in some cases.

2 Background: Morphological Inflection

2.1 Inflectional Morphology

To properly discuss models of inflectional morphology, we require a formalization. We adopt the framework of word-based morphology (Aronoff, 1976; Spencer, 1991). Note in the present paper, we omit derivational morphology.

We define an **inflected lexicon** as a set of 4-tuples consisting of a part-of-speech tag, a lexeme, an inflectional slot, and a surface form. A **lexeme** is a discrete object that indexes the word's core meaning and part of speech. In place of such an abstract lexeme, lexicographers will often use a **lemma**, denoted by ℓ, which is a designated[2] surface form of the lexeme (such as the infinitive). For the remainder of this paper, we will use the lemma as a proxy for the lexeme, wherever convenient, although we note that lemmata may be ambiguous: bank is the lemma for at least two distinct nouns and two distinct verbs. For inflection, this ambiguity will rarely[3] play a role—for instance, all senses of bank inflect in the same fashion.

A **part-of-speech (POS) tag**, denoted t, is a coarse syntactic category such as VERB. Each POS tag allows some set of lexemes, and also allows some set of inflectional **slots**, denoted as σ, such as $\left[\text{TNS}{=}\text{PAST}, \text{PERSON}{=}3\right]$. Each allowed $\langle$tag, lexeme, slot$\rangle$ triple is realized—in only one way—as an inflected **surface form**, a string over a fixed phonological or orthographic alphabet Σ. (In this work, we take Σ to be an orthographic alphabet.) Additionally, we will define the term **morphological tag**, denoted by m, which we take to be the POS-slot pair $m = \langle t, \sigma \rangle$. We will further define $\mathcal{T}$ as the set of all POS tags and $\mathcal{M}$ as the set of all morphological tags.

A **paradigm** $\pi(t, \ell)$ is the mapping from tag t's slots to the surface forms that "fill" those slots for lexeme/lemma ℓ. For example, in the English paradigm $\pi(\text{VERB}, \text{talk})$, the past-tense slot is said to be filled by talked, meaning that the lexicon contains the tuple $\langle \text{VERB}, \text{talk}, \text{PAST}, \text{talked} \rangle$.

A cheat sheet for the notation is provided in Tab. 2.

We will specifically work with the UniMorph annotation scheme (Sylak-Glassman, 2016). Here, each slot specifies a morpho-syntactic bundle of inflectional features such as tense, mood, person, number, and gender. For example, the German surface form Wörtern is listed in the lexicon with tag NOUN, lemma Wort, and a slot specifying the feature bundle $\left[\text{NUM}{=}\text{PL}, \text{CASE}{=}\text{DAT}\right]$. The full paradigms $\pi(\text{NOUN}, \text{Wort})$ and $\pi(\text{NOUN}, \text{Herr})$ are found in Tab. 1.

2.2 Morphological Inflection

Now, we formulate the task of context-free **morphological inflection** using the notation developed in §2. Given a set of N form-tag-lemma triples

[1] We make our code and data available at: https://github.com/lwolfsonkin/morph-svae.

[2] A specific slot of the paradigm is chosen, depending on the part-of-speech tag – all these terms are defined next.

[3] One example of a paradigm where the lexeme, rather than the lemma, may influence inflection is hang. If one chooses the lexeme that licenses animate objects, the proper past tense is hanged, whereas it is hung for the lexeme that licenses inanimate objects.

object	symbol	example
form	f	talking
lemma	ℓ	talk
POS	t	VERB
slot	σ	[TNS=GERUND]
morph. tag	m	[POS=V, TNS=GERUND]

Table 2: Notational cheat sheet for the paper.

$\{\langle f_i, m_i, \ell_i \rangle\}_{i=1}^{N}$, the goal of morphological inflection is to map the pair $\langle m_i, \ell_i \rangle$ to the form f_i. As the definition above indicates, the task is traditionally performed at the *type level*. In this work, however, we focus on a generalization of the task to the *token level*—we seek to map a bisequence of lemma-tag pairs to the sequence of inflected forms in context. Formally, we will denote the lemma-morphological tag bisequence as $\langle \boldsymbol{\ell}, \boldsymbol{m} \rangle$ and the form sequence as $\boldsymbol{f}$. Foreshadowing, the primary motivation for this generalization is to enable the use of raw-text in a semi-supervised setting.

3 Generating Sequences of Inflections

The primary contribution of this paper is a novel generative model over sequences of inflected words in their sentential context. Following the notation laid out in §2.2, we seek to jointly learn a distribution over sequences of forms $\boldsymbol{f}$, lemmata $\boldsymbol{\ell}$, and morphological tags $\boldsymbol{m}$. The generative procedure is as follows: First, we sample a sequence of tags $\boldsymbol{m}$, each morphological tag coming from a language model over morphological tags: $m_i \sim p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{m}_{<i})$. Next, we sample the sequence of lemmata $\boldsymbol{\ell}$ given the previously sampled sequence of tags $\boldsymbol{m}$— these are sampled conditioned only on the corresponding morphological tag: $\ell_i \sim p_{\boldsymbol{\theta}}(\cdot \mid m_i)$. Finally, we sample the sequence of inflected words $\boldsymbol{f}$, where, again, each word is chosen conditionally independent of other elements of the sequence: $f_i \sim p_{\boldsymbol{\theta}}(\cdot \mid \ell_i, m_i)$.[4] This yields the factorized joint distribution:

$$p_{\boldsymbol{\theta}}(\boldsymbol{f}, \boldsymbol{\ell}, \boldsymbol{m}) = \tag{1}$$
$$\left(\prod_{i=1}^{|\boldsymbol{f}|} \underbrace{p_{\boldsymbol{\theta}}(f_i \mid \ell_i, m_i)}_{\textit{morphological inflector}\ \textcircled{3}} \cdot \underbrace{p_{\boldsymbol{\theta}}(\ell_i \mid m_i)}_{\textit{lemma generator}\ \textcircled{2}} \right) \cdot \underbrace{p_{\boldsymbol{\theta}}(\boldsymbol{m})}_{\textit{m-tag LM}\ \textcircled{1}}$$

[4]Note that we denote all three distributions as $p_{\boldsymbol{\theta}}$ to simplify notation and emphasize the joint modeling aspect; context will always resolve the ambiguity in this paper. We will discuss their parameterization in §4.

We depict the corresponding directed graphical model in Fig. 1.

Relation to Other Models in NLP. As the graphical model drawn in Fig. 1 shows, our model is quite similar to a Hidden Markov Model (HMM) (Rabiner, 1989). There are two primary differences. First, we remark that an HMM directly emits a form f_i conditioned on the tag m_i. Our model, in contrast, emits a lemma ℓ_i conditioned on the morphological tag m_i and, then, conditioned on both the lemma ℓ_i and the tag m_i, we emit the inflected form f_i. In this sense, our model resembles the hierarchical HMM of Fine et al. (1998) with the difference that we do not have interdependence between the lemmata ℓ_i. The second difference is that our model is non-Markovian: we sample the i^{th} morphological tag m_i from a distribution that depends on *all* previous tags, using an LSTM language model (§4.1). This yields richer interactions among the tags, which may be necessary for modeling long-distance agreement phenomena.

Why a Generative Model? What is our interest in a generative model of inflected forms? Eq. (1) is a syntax-only language model in that it only allows for interdependencies between the morpho-syntactic tags in $p_{\boldsymbol{\theta}}(\boldsymbol{m})$. However, given a tag sequence $\boldsymbol{m}$, the individual lemmata and forms are *conditionally independent*. This prevents the model from learning notions such as semantic frames and topicality. So what is this model good for? Our chief interest is the ability to *train a morphological inflector on unlabeled data*, which is a boon in a low-resource setting. As the model is generative, we may consider the latent-variable model:

$$p_{\boldsymbol{\theta}}(\boldsymbol{f}) = \sum_{\langle \boldsymbol{\ell}, \boldsymbol{m} \rangle} p_{\boldsymbol{\theta}}(\boldsymbol{f}, \boldsymbol{\ell}, \boldsymbol{m}), \tag{2}$$

where we marginalize out the latent lemmata and morphological tags from raw text. The sum in Eq. (2) is unabashedly intractable—given a sequence $\boldsymbol{f}$, it involves consideration of an exponential (in $|\boldsymbol{f}|$) number of tag sequences and an *infinite* number of lemmata sequences. Thus, we will fall back on an approximation scheme (see §5).

4 Recurrent Neural Parameterization

The graphical model from §3 specifies a family of models that obey the conditional independence assumptions dictated by the graph in Fig. 1. In this section we define a specific parameterization using

long short-term memory (LSTM) recurrent neural network (Hochreiter and Schmidhuber, 1997) language models (Sundermeyer et al., 2012).

4.1 LSTM Language Models

Before proceeding, we review the modeling of sequences with LSTM language models. Given some alphabet Δ, the distribution over sequences $\mathbf{x} \in \Delta^*$ can be defined as follows:

$$p(\mathbf{x}) = \prod_{j=1}^{|\mathbf{x}|} p(x_j \mid \mathbf{x}_{<j}), \tag{3}$$

where $\mathbf{x}_{<j} = x_1, \ldots, x_{j-1}$. The prediction at time step j of a single element x_j is then parametrized by a neural network:

$$p(x_j \mid \mathbf{x}_{<j}) = \text{softmax}\left(\mathbf{W} \cdot \mathbf{h}_j + \mathbf{b}\right), \tag{4}$$

where $\mathbf{W} \in \mathbb{R}^{|\Delta| \times d}$ and $\mathbf{b} \in \mathbb{R}^{|\Delta|}$ are learned parameters (for some number of hidden units d) and the hidden state $\mathbf{h}_j \in \mathbb{R}^d$ is defined through the recurrence given by Hochreiter and Schmidhuber (1997) from the previous hidden state and an embedding of the previous character (assuming some learned embedding function $\mathbf{e}\colon \Delta \to \mathbb{R}^c$ for some number of dimensions c):

$$\mathbf{h}_j = \text{LSTM}\big(\mathbf{h}_{j-1}, \mathbf{e}(x_{j-1})\big) \tag{5}$$

4.2 Our Conditional Distributions

We discuss each of the factors in Eq. (1) in turn.

① **Morphological Tag Language Model: $p_{\boldsymbol{\theta}}(\boldsymbol{m})$.** We define $p_{\boldsymbol{\theta}}(\boldsymbol{m})$ as an LSTM language model, as defined in §4.1, where we take $\Delta = \mathcal{M}$, i.e., the elements of the sequence that are to be predicted are tags like $\big[\text{POS=V}, \text{TNS=GERUND}\big]$. Note that the embedding function $\mathbf{e}$ does not treat them as atomic units, but breaks them up into individual attribute-value pairs that are embedded individually and then summed to yield the final vector representation. To be precise, each tag is first encoded by a multi-hot vector, where each component corresponds to a attribute-value pair in the slot, and then this multi-hot vector is multiplied with an embedding matrix.

② **Lemma Generator: $p_{\boldsymbol{\theta}}(\ell_i \mid m_i)$.** The next distribution in our model is a lemma generator which we define to be a *conditional* LSTM language model over characters (we take $\Delta = \Sigma$), i.e., each x_i is a single (orthographic) character. The language model is conditioned on t_i (the part-of-speech information contained in the morphological tag $m_i = \langle t_i, \sigma_i \rangle$), which we embed into a low-dimensional space and feed to the LSTM by concatenating its embedding with that of the current character. Thusly, we obtain the new recurrence relation for the hidden state:

$$\mathbf{h}_j = \text{LSTM}\Big(\mathbf{h}_{j-1}, \Big[\, \mathbf{e}\big([\ell_i]_{j-1}\big) \,;\, \mathbf{e}'\big(t_i\big) \,\Big]\Big), \tag{6}$$

where $[\ell_i]_j$ denotes the j^{th} character of the generated lemma ℓ_i and $\mathbf{e}' : \mathcal{T} \to \mathbb{R}^{c'}$ for some c' is a learned embedding function for POS tags. Note that we embed only the POS tag, rather than the entire morphological tag, as we assume the lemma depends on the part of speech exclusively.

③ **Morphological Inflector: $p_{\boldsymbol{\theta}}(f_i \mid \ell_i, m_i)$.** The final conditional in our model is a morphological inflector, which we parameterize as a neural recurrent sequence-to-sequence model (Sutskever et al., 2014) with Luong dot-style attention (Luong et al., 2015). Our particular model uses a single encoder-decoder architecture (Kann and Schütze, 2016) for all tag pairs within a language and we refer to reader to that paper for further details. Concretely, the encoder runs over a string consisting of the desired slot and all characters of the lemma that is to be inflected (e.g. `<w> V PST t a l k </w>`), one LSTM running left-to-right, the other right-to-left. Concatenating the hidden states of both RNNs at each time step results in hidden states $\mathbf{h}_j^{(enc)}$. The decoder, again, takes the form of an LSTM language model (we take $\Delta = \Sigma$), producing the inflected form character by character, but at each time step not only the previous hidden state and the previously generated token are considered, but attention (a convex combination) over all encoder hidden states $\mathbf{h}_j^{(\text{enc})}$, with the distribution given by another neural network; see Luong et al. (2015).

5 Semi-Supervised Wake-Sleep

We train the model with the wake-sleep procedure, which requires us to perform posterior inference over the latent variables. However, the exact computation in the model is intractable—it involves a sum over all possible lemmatizations and taggings of the sentence, as shown in Eq. (2). Thus, we fall back on a variational approximation (Jordan et al., 1999). We train an **inference network** $q_{\boldsymbol{\phi}}(\boldsymbol{\ell}, \boldsymbol{m} \mid \boldsymbol{f})$ that approximates the true posterior over the latent variables $p_{\boldsymbol{\theta}}(\boldsymbol{\ell}, \boldsymbol{m} \mid \boldsymbol{f})$.[5] The

[5] Inference networks are also known as **stochastic inverses** (Stuhlmüller et al., 2013) or **recognition models** (Dayan et al.,

variational family we choose in this work will be detailed in §5.5. We fit the distribution q_{ϕ} using a semi-supervised extension of the wake-sleep algorithm (Hinton et al., 1995; Dayan et al., 1995; Bornschein and Bengio, 2014). We derive the algorithm in the following subsections and provide pseudo-code in Alg. 1.

Note that the wake-sleep algorithm shows structural similarities to the expectation-maximization (EM) algorithm (Dempster et al., 1977), and, presaging the exposition, we note that the wake-sleep procedure is a type of variational EM (Beal, 2003). The key difference is that the E-step minimizes an inclusive KL divergence, rather than the exclusive one typically found in variational EM.

5.1 Data Requirements of Wake-Sleep

We emphasize again that we will train our model in a semi-supervised fashion. Thus, we will assume a set of labeled sentences, $\mathcal{D}_{\textit{labeled}}$, represented as a set of triples $\langle \boldsymbol{f}, \boldsymbol{\ell}, \boldsymbol{m} \rangle$, and a set of unlabeled sentences, $\mathcal{D}_{\textit{unlabeled}}$, represented as a set of surface form sequences $\boldsymbol{f}$.

5.2 The Sleep Phase

Wake-sleep first dictates that we find an approximate posterior distribution q_{ϕ} that minimizes the KL divergences for all form sequences:

$$D_{KL}\Big(\underbrace{p_{\boldsymbol{\theta}}(\cdot, \cdot, \cdot)}_{\textit{full joint: Eq. (1)}} \;\Big|\Big|\; \underbrace{q_{\boldsymbol{\phi}}(\cdot, \cdot \mid \cdot)}_{\textit{variational approximation}} \Big) \quad (7)$$

with respect to the parameters $\boldsymbol{\phi}$, which control the variational approximation $q_{\boldsymbol{\phi}}$. Because $q_{\boldsymbol{\phi}}$ is trained to be a variational approximation for *any* input $\boldsymbol{f}$, it is called an inference network. In other words, it will return an approximate posterior over the latent variables for any observed sequence. Importantly, note that computation of Eq. (7) is still hard—it requires us to normalize the distribution $p_{\boldsymbol{\theta}}$, which, in turn, involves a sum over all lemmatizations and taggings. However, it does lend itself to an efficient Monte Carlo approximation. As our model is fully generative and directed, we may easily take samples from the complete joint. Specifically, we will take K samples $\langle \tilde{\boldsymbol{f}}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle \sim p_{\boldsymbol{\theta}}(\cdot, \cdot, \cdot)$ by forward sampling and define them as $\widetilde{\mathcal{D}}_{\textit{sleep}}$. We remark that we use a tilde to indicate that a form, lemmata or tag is sampled, rather than human annotated. Using K samples, we obtain the objective

$$\mathcal{S}_{\textit{unsup}} = {}^{1}\!/_{K} \cdot \sum_{\langle \tilde{\boldsymbol{f}}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle \in \widetilde{\mathcal{D}}_{\textit{sleep}}} \log q_{\boldsymbol{\phi}}(\tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \mid \tilde{\boldsymbol{f}}), \quad (8)$$

which we could maximize by fitting the model $q_{\boldsymbol{\phi}}$ through backpropagation (Rumelhart et al., 1986), as one would during maximum likelihood estimation.

5.3 The Wake Phase

Now, given our approximate posterior $q_{\boldsymbol{\phi}}(\boldsymbol{\ell}, \boldsymbol{m} \mid \boldsymbol{f})$, we are in a position to re-estimate the parameters of the generative model $p_{\boldsymbol{\theta}}(\boldsymbol{f}, \boldsymbol{\ell}, \boldsymbol{m})$. Given a set of unannotated sentences $\mathcal{D}_{\textit{unlabeled}}$, we again first consider the objective

$$\mathcal{W}_{\textit{unsup}} = {}^{1}\!/_{M} \cdot \sum_{\langle \boldsymbol{f}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle \in \widetilde{\mathcal{D}}_{\textit{wake}}} \log p_{\boldsymbol{\theta}}(\boldsymbol{f}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}}) \quad (9)$$

where $\widetilde{\mathcal{D}}_{\textit{wake}}$ is a set of triples $\langle \boldsymbol{f}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle$ with $\boldsymbol{f} \in \mathcal{D}_{\textit{unlabeled}}$ and $\langle \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle \sim q_{\boldsymbol{\phi}}(\cdot, \cdot \mid \boldsymbol{f})$, maximizing with respect to the parameters $\boldsymbol{\theta}$ (we may stochastically backprop through the expectation simply by backpropagating through this sum). Note that Eq. (9) is a Monte Carlo approximation of the inclusive divergence of the data distribution of $\mathcal{D}_{\textit{unlabeled}}$ times $q_{\boldsymbol{\phi}}$ with $p_{\boldsymbol{\theta}}$.

5.4 Adding Supervision to Wake-Sleep

So far we presented a purely unsupervised training method that makes no assumptions about the latent lemmata and morphological tags. In our case, however, we have a very clear idea what the latent variables should look like. For instance, we are quite certain that the lemma of talking is talk and that it is in fact a GERUND. And, indeed, we have access to annotated examples $\mathcal{D}_{\textit{labeled}}$ in the form of an annotated corpus. In the presence of these data, we optimize the supervised sleep phase objective,

$$\mathcal{S}_{\textit{sup}} = {}^{1}\!/_{N} \cdot \sum_{\langle \boldsymbol{f}, \boldsymbol{\ell}, \boldsymbol{m} \rangle \in \mathcal{D}_{\textit{labeled}}} \log q_{\boldsymbol{\phi}}(\boldsymbol{\ell}, \boldsymbol{m} \mid \boldsymbol{f}). \quad (10)$$

which is a Monte Carlo approximation of $D_{KL}(\mathcal{D}_{\textit{labeled}} \;||\; q_{\boldsymbol{\phi}})$. Thus, when fitting our variational approximation $q_{\boldsymbol{\phi}}$, we will optimize a joint objective $\mathcal{S} = \mathcal{S}_{\textit{sup}} + \gamma_{\textit{sleep}} \cdot \mathcal{S}_{\textit{unsup}}$, where $\mathcal{S}_{\textit{sup}}$, to repeat, uses actual annotated lemmata and morphological tags; we balance the two parts of the objective with a scaling parameter $\gamma_{\textit{sleep}}$. Note that on the first sleep phase iteration, we set $\gamma_{\textit{sleep}} = 0$ since taking samples from an untrained $p_{\boldsymbol{\theta}}(\cdot, \cdot, \cdot)$

1995).

Algorithm 1 semi-supervised wake-sleep

input $\mathcal{D}_{labeled}$ ▷ *labeled training data*
input $\mathcal{D}_{unlabeled}$ ▷ *unlabeled training data*
for $i = 1$ **to** I **do**
  $\widetilde{\mathcal{D}}_{sleep} \leftarrow \emptyset$
  if $i > 1$ **then**
    for $k = 1$ **to** K **do**
      $\langle \tilde{\boldsymbol{f}}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle \sim p_{\boldsymbol{\theta}}(\cdot, \cdot, \cdot)$
      $\widetilde{\mathcal{D}}_{sleep} \leftarrow \widetilde{\mathcal{D}}_{sleep} \cup \{\langle \tilde{\boldsymbol{f}}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle\}$
  maximize $\log q_{\boldsymbol{\phi}}$ on $\mathcal{D}_{labeled} \cup \widetilde{\mathcal{D}}_{sleep}$
    ▷ *this corresponds to Eq. (10) + Eq. (8)*
  $\widetilde{\mathcal{D}}_{wake} \leftarrow \emptyset$
  for $\boldsymbol{f} \in \mathcal{D}_{unlabeled}$ **do**
    $\langle \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle \sim q_{\boldsymbol{\phi}}(\cdot, \cdot \mid \boldsymbol{f})$
    $\widetilde{\mathcal{D}}_{wake} \leftarrow \widetilde{\mathcal{D}}_{wake} \cup \{\langle \boldsymbol{f}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle\}$
  maximize $\log p_{\boldsymbol{\theta}}$ on $\mathcal{D}_{labeled} \cup \widetilde{\mathcal{D}}_{wake}$
    ▷ *this corresponds to Eq. (11) + Eq. (9)*

when we have available labeled data is of little utility. We will discuss the provenance of our data in §7.2.

Likewise, in the wake phase we can neglect the approximation $q_{\boldsymbol{\phi}}$ in favor of the annotated latent variables found in $\mathcal{D}_{labeled}$; this leads to the following supervised objective

$$\mathcal{W}_{sup} = {}^1\!/_N \cdot \sum_{\langle \boldsymbol{f}, \boldsymbol{\ell}, \boldsymbol{m} \rangle \in \mathcal{D}_{labeled}} \log p_{\boldsymbol{\theta}}(\boldsymbol{f}, \boldsymbol{\ell}, \boldsymbol{m}), \quad (11)$$

which is a Monte Carlo approximation of $D_{KL}(\mathcal{D}_{labeled} \mid\mid p_{\boldsymbol{\theta}})$. As in the sleep phase, we will maximize $\mathcal{W} = \mathcal{W}_{sup} + \gamma_{wake} \cdot \mathcal{W}_{unsup}$, where γ_{wake} is, again, a scaling parameter.

5.5 Our Variational Family

How do we choose the variational family $q_{\boldsymbol{\phi}}$? In terms of NLP nomenclature, $q_{\boldsymbol{\phi}}$ represents a joint morphological tagger and lemmatizer. The open-source tool LEMMING (Müller et al., 2015) represents such an object. LEMMING is a higher-order linear-chain conditional random field (CRF; Lafferty et al., 2001), that is an extension of the morphological tagger of Müller et al. (2013). Interestingly, LEMMING is a linear model that makes use of simple character n-gram feature templates. On both the tasks of morphological tagging and lemmatization, neural models have supplanted linear models in terms of performance in the high-resource case (Heigold et al., 2017). However, we are interested in producing an accurate approximation to the posterior in the presence of minimal annotated examples and potentially noisy samples produced during the sleep phase, where linear models still outperform non-linear approaches (Cotterell and Heigold, 2017). We note that our variational approximation is compatible with any family.

5.6 Interpretation as an Autoencoder

We may also view our model as an autoencoder, following Kingma and Welling (2013), who saw that a variational approximation to any generative model naturally has this interpretation. The crucial difference between Kingma and Welling (2013) and this work is that our model is a **structured** variational autoencoder in the sense that the space of our latent code is structured: the inference network encodes a sentence into a pair of lemmata and morphological tags $\langle \boldsymbol{\ell}, \boldsymbol{m} \rangle$. This bisequence is then decoded back into the sequence of forms $\boldsymbol{f}$ through a morphological inflector. The reason the model is called an autoencoder is that we arrive at an auto-encoding-like objective if we combine the $p_{\boldsymbol{\theta}}$ and $q_{\boldsymbol{\phi}}$ as so:

$$p(\boldsymbol{f} \mid \hat{\boldsymbol{f}}) = \sum_{\langle \boldsymbol{\ell}, \boldsymbol{m} \rangle} p_{\boldsymbol{\theta}}(\boldsymbol{f} \mid \boldsymbol{\ell}, \boldsymbol{m}) \cdot q_{\boldsymbol{\phi}}(\boldsymbol{\ell}, \boldsymbol{m} \mid \hat{\boldsymbol{f}}) \quad (12)$$

where $\hat{\boldsymbol{f}}$ is a copy of the original sentence $\boldsymbol{f}$.

Note that this choice of latent space sadly precludes us from making use of the *reparametrization trick* that makes inference in VAEs particularly efficient. In fact, our whole inference procedure is quite different as we do not perform gradient descent on both $q_{\boldsymbol{\phi}}$ and $p_{\boldsymbol{\theta}}$ jointly but alternatingly optimize both (using wake-sleep). We nevertheless call our model a VAE to uphold the distinction between the VAE as a model (essentially a specific Helmholtz machine (Dayan et al., 1995), justified by variational inference) and the end-to-end inference procedure that is commonly used.

Another way of viewing this model is that it tries to force the words in the corpus through a syntactic bottleneck. Spiritually, our work is close to the conditional random field autoencoder of Ammar et al. (2014).

We remark that many other structured NLP tasks can be "autoencoded" in this way and, thus, trained by a similar wake-sleep procedure. For instance, any two tasks that effectively function as inverses, e.g., translation and backtranslation, or language generation and parsing, can be treated with a similar variational autoencoder. While this work only

focuses on the creation of an improved morphological inflector $p_{\theta}(\boldsymbol{f} \mid \boldsymbol{\ell}, \boldsymbol{m})$, one could imagine a situation where the encoder was also a task of interest. That is, the goal would be to improve both the decoder (the generation model) *and* the encoder (the variational approximation).

6 Related Work

Closest to our work is Zhou and Neubig (2017), who describe an unstructured variational autoencoder. However, the exact use case of our respective models is distinct. Our method models the syntactic dynamics with an LSTM language model over morphological tags. Thus, in the semi-supervised setting, we require token-level annotation. Additionally, our latent variables are interpretable as they correspond to well-understood linguistic quantities. In contrast, Zhou and Neubig (2017) infer latent lemmata as real vectors. To the best of our knowledge, we are only the second attempt, after Zhou and Neubig (2017), to attempt to perform semi-supervised learning for a neural inflection generator. Other non-neural attempts at semi-supervised learning of morphological inflectors include Hulden et al. (2014). Models in this vein are non-neural and often focus on exploiting corpus statistics, e.g., token frequency, rather than explicitly modeling the forms in context. All of these approaches are designed to learn from a type-level lexicon, rendering direct comparison difficult.

7 Experiments

While we estimate all the parameters in the generative model, the purpose of this work is to improve the performance of morphological inflectors through semi-supervised learning with the incorporation of unlabeled data.

7.1 Low-Resource Inflection Generation

The development of our method was primarily aimed at the low-resource scenario, where we observe a limited number of annotated data points. Why low-resource? When we have access to a preponderance of data, morphological inflection is close to being a solved problem, as evinced in SIGMORPHON's 2016 shared task. However, the CoNLL-SIGMORPHON 2017 shared task showed there is much progress to be made in the low-resource case. Semi-supervision is a clear avenue.

7.2 Data

As our model requires *token-level* morphological annotation, we perform our experiments on the Universal Dependencies (UD) dataset (Nivre et al., 2017). As this stands in contrast to most work on morphological inflection (which has used the UniMorph (Sylak-Glassman et al., 2015)[6] datasets), we use a converted version of UD data, in which the UD morphological tags have been deterministically converted into UniMorph tags.

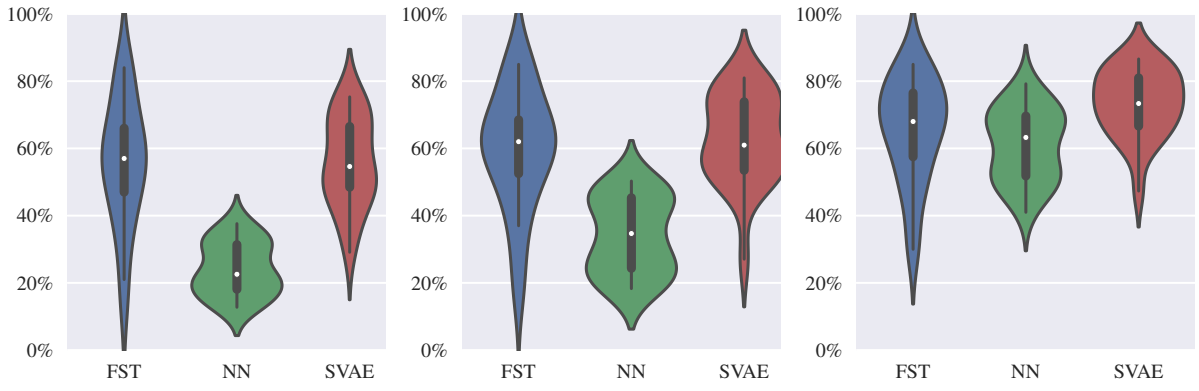
For each of the treebanks in the UD dataset, we divide the training portion into three chunks consisting of the first 500, 1000 and 5000 tokens, respectively. These *labeled* chunks will constitute three unique sets $\mathcal{D}_{labeled}$. The remaining sentences in the training portion will be used as *unlabeled* data $\mathcal{D}_{unlabeled}$ for each language, i.e., we will discard those labels. The development and test portions will be left untouched.

Languages. We explore a typologically diverse set of languages of various stocks: Indo-European, Afro-Asiatic, Turkic and Finno-Ugric, as well as the language isolate Basque. We have organized our experimental languages in Tab. 3 by genetic grouping, highlighting sub-families where possible. The Indo-European languages mostly exhibit fusional morphologies of varying degrees of complexity. The Basque, Turkic, and Finno-Ugric languages are agglutinative. Both of the Afro-Asiatic languages, Arabic and Hebrew, are Semitic and have templatic morphology with fusional affixes.

7.3 Evaluation

The end product of our procedure is a morphological inflector, whose performance is to be improved through the incorporation of unlabeled data. Thus, we evaluate using the standard metric accuracy. We will evaluate at the *type level*, as is traditional in the morphological inflection literature, even though the UD treebanks on which we evaluate are *token-level* resources. Concretely, we compile an incomplete type-level morphological lexicon from the token-level resource. To create this resource, we gather all unique form-lemma-tag triples $\langle f, \ell, m \rangle$ present

[6] The two annotation schemes are similar. For a discussion, we refer the reader to http://universaldependencies.org/v2/features.html; sadly there are differences that render all numbers reported in this work incomparable with previous work, see §7.4.



(a) 500 training tokens (b) 1000 training tokens (c) 5000 training tokens

Figure 2: Violin plots showing the distribution over accuracies. The structured variational autoencoder (SVAE) always outperforms the neural network (NN), but only outperformed the FST-based approach when trained on 5000 annotated tokens. Thus, while semi-supervised training helps neural models reduce their sample complexity, roughly 5000 annotated tokens are still required to boost their performance above more symbolic baselines.

in the UD test data.[7]

7.4 Baselines

As mentioned before, most work on morphological inflection has considered the task of estimating statistical inflectors from type-level lexicons. Here, in contrast, we require token-level annotation to estimate our model. For this reason, there is neither a competing approach whose numbers we can make a fair comparison to nor is there an open-source system we could easily run in the token-level setting. This is why we treat our token-level data as a list of "types"[8] and then use two simple type-based baselines.

First, we consider the probabilistic finite-state transducer used as the baseline for the CoNLL-SIGMORPHON 2017 shared task.[9] We consider this a relatively strong baseline, as we seek to generalize from a minimal amount of data. As described by Cotterell et al. (2017), the baseline performed quite competitively in the task's low-resource setting. Note that the finite-state machine is created by heuristically extracting prefixes and suffixes from the word forms, based on an unsupervised alignment step. The second baseline is our neural inflector $p(f \mid \ell, m)$ given in §4 *without* the semi-supervision; this model is state-of-the-art on the high-resource version of the task.

We will refer to our baselines as follows: **FST** is the probabilistic transducer, **NN** is the neural sequence-to-sequence model *without* semi-supervision, and **SVAE** is the structured variational autoencoder, which is equivalent to **NN** but also trained using wake-sleep and unlabeled data.

7.5 Results

We ran the three models on 23 languages with the hyperparameters and experimental details described in App. A. We present our results in Fig. 2 and in Tab. 3. We also provide sample output of the generative model created using the dream step in App. B. The high-level take-away is that on almost all languages we are able to exploit the unlabeled data to improve the sequence-to-sequence model using unlabeled data, i.e., SVAE outperforms the NN model on *all* languages across *all* training scenarios. However, in many cases, the FST model is a better choice—the FST can sometimes generalize better from a handful of supervised examples than the neural network, even with semi-supervision (SVAE). We highlight three finer-grained observations below.

Observation 1: FST Good in Low-Resource. As clearly evinced in Fig. 2, the baseline FST is still competitive with the NN, or even our SVAE when data is extremely scarce. Our neural architecture is quite general, and lacks the prior knowledge and inductive biases of the rule-based system, which become more pertinent in low-resource scenarios. Even though our semi-supervised strategy clearly

[7] Some of these form-lemma-tag triples will overlap with those seen in the training data.

[8] Typical type-based inflection lexicons are likely not i.i.d. samples from natural utterances, but we have no other choice if we want to make use of only our token-level data and not additional resources like frequency and regularity of forms.

[9] https://sites.google.com/view/conll-sigmorphon2017/

			500 tokens					1000 tokens					5000 tokens				
		lang	FST	NN	SVAE	Δ_{FST}	Δ_{NN}	FST	NN	SVAE	Δ_{FST}	Δ_{NN}	FST	NN	SVAE	Δ_{FST}	Δ_{NN}
Romance		ca	81.0	28.11	71.76	-9.24	43.65	85.0	42.58	78.46	-6.54	35.88	84.0	74.22	85.77	1.77	11.55
		fr	84.0	36.25	74.75	-9.25	38.5	85.0	47.04	79.97	-5.03	32.93	85.0	79.21	83.96	-1.04	4.75
		it	81.0	31.30	67.48	-13.52	36.18	81.0	43.58	77.37	-3.63	33.79	82.0	71.09	73.11	-8.89	2.02
		la	21.0	14.02	29.12	8.12	15.10	26.0	19.62	27.06	1.06	7.44	30.0	41.00	47.32	17.32	6.32
		pt	81.0	31.58	72.54	-8.46	40.96	83.0	47.27	73.24	-9.76	25.97	82.0	74.17	86.13	4.13	11.96
		ro	56.0	22.56	52.48	-3.52	29.92	62.0	34.68	58.30	-3.70	23.62	68.0	51.77	75.49	7.49	23.72
		es	57.0	34.34	75.32	18.32	40.98	60.0	46.14	80.97	20.97	34.83	72.0	71.99	84.44	12.44	12.45
Germanic		nl	63.0	19.22	49.14	-13.86	29.92	65.0	26.05	53.12	-11.88	27.07	70.0	53.70	65.97	-4.03	12.27
		da	68.0	31.25	65.58	-2.42	34.33	73.0	44.51	72.82	-0.18	28.31	79.0	67.92	80.12	1.12	12.20
		no	69.0	32.51	65.46	-3.54	32.95	71.0	46.26	74.49	3.49	28.23	79.0	71.31	81.25	2.25	9.94
		nn	64.0	20.29	54.62	-9.38	34.33	65.0	24.32	60.97	-4.03	36.65	72.0	50.40	73.35	1.35	22.95
		sv	63.0	19.02	58.15	-4.85	39.13	66.0	36.35	67.18	1.18	30.83	74.0	59.82	78.23	4.23	18.41
Slavic		bg	44.0	15.51	47.22	3.22	31.71	51.0	21.00	57.18	6.18	36.18	59.0	49.06	71.15	12.15	22.09
		pl	50.0	12.75	48.62	-1.38	35.87	57.0	19.88	55.90	-1.10	36.02	64.0	54.44	67.15	3.15	12.71
		si	52.0	15.60	55.69	3.69	40.09	61.0	26.39	61.22	0.22	34.83	68.0	66.65	75.40	7.40	8.75
Semit.		ar	14.0	31.47	63.53	49.53	32.06	17.0	48.53	71.52	54.52	22.99	34.0	68.16	80.72	46.72	12.56
		he	60.0	37.61	71.11	11.11	33.50	66.0	50.28	76.32	10.32	26.04	72.0	64.37	86.60	14.6	22.23
Finn.-Urg.		hu	53.0	22.56	48.64	-4.36	26.08	56.0	28.62	60.74	4.74	32.12	61.0	66.45	72.84	11.84	6.39
		et	39.0	21.81	42.16	3.16	20.35	45.0	29.66	51.75	6.75	22.09	49.0	46.82	58.91	9.91	12.09
		fi	37.0	12.97	35.78	-1.22	22.81	42.0	19.03	47.65	5.65	28.62	49.0	46.75	62.76	13.76	16.01
other		lv	57.0	17.16	48.29	-8.71	31.13	63.0	18.30	53.58	-9.42	35.28	66.0	51.84	66.12	0.12	14.28
		eu	50.0	24.46	48.72	-1.28	24.26	54.0	35.14	53.39	-0.61	18.25	56.0	56.29	62.33	6.33	6.04
		tr	34.0	20.67	37.92	3.92	17.25	37.0	24.33	49.67	12.67	25.34	48.0	63.26	69.35	21.35	6.09
		avg	55.57	24.04	**55.83**	0.26	31.79	59.61	33.89	**62.73**	3.12	6.90	65.35	60.90	**73.41**	8.06	12.51

Table 3: Type-level morphological inflection accuracy across different models, training scenarios, and languages

improves the performance of NN, we cannot always recommend SVAE for the case when we only have 500 annotated tokens, but on average it does slightly better. The SVAE surpasses the FST when moving up to 1000 annotated tokens, becoming even more pronounced at 5000 annotated tokens.

Observation 2: Agglutinative Languages. The next trend we remark upon is that languages of an agglutinating nature tend to benefit more from the semi-supervised learning. Why should this be? Since in our experimental set-up, every language sees the *same number of tokens*, it is naturally harder to generalize on languages that have more distinct morphological variants. Also, by the nature of agglutinative languages, relevant morphemes could be arbitrarily far from the edges of the string, making the (NN and) SVAE's ability to learn more generic rules even more valuable.

Observation 3: Non-concatenative Morphology. One interesting advantage that the neural models have over the FSTs is the ability to learn non-concatenative phenomena. The FST model is based on prefix and suffix rewrite rules and, naturally, struggles when the correctly reinflected form is more than the concatenation of these parts. Thus we see that for the two semitic language, the SVAE is the best method across all resource settings.

8 Conclusion

We have presented a novel generative model for morphological inflection generation in context. The model allows us to exploit unlabeled data in the training of morphological inflectors. As the model's rich parameterization prevents tractable inference, we craft a variational inference procedure, based on the wake-sleep algorithm, to marginalize out the latent variables. Experimentally, we provide empirical validation on 23 languages. We find that, especially in the lower-resource conditions, our model improves by large margins over the baselines.

References

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

Waleed Ammar, Chris Dyer, and Noah A. Smith. 2014. Conditional random field autoencoders for unsupervised structured prediction. In *Advances in Neural Information Processing Systems*, pages 3311–3319.

Mark Aronoff. 1976. *Word Formation in Generative Grammar*. Number 1 in Linguistic Inquiry Monographs. MIT Press, Cambridge, MA.

Matthew James Beal. 2003. *Variational Algorithms for Approximate Bayesian Inference*. University College London.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Jörg Bornschein and Yoshua Bengio. 2014. Reweighted wake-sleep. *CoRR*, abs/1406.2751.

Ryan Cotterell and Georg Heigold. 2017. Cross-lingual character-level neural morphological tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 748–759, Copenhagen, Denmark. 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. The CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL-SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, Vancouver, Canada. 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.

Peter Dayan, Geoffrey E. Hinton, Radford M. Neal, and Richard S. Zemel. 1995. The Helmholtz machine. *Neural Computation*, 7(5):889–904.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

Markus Dreyer, Jason Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1080–1089, Honolulu, Hawaii. Association for Computational Linguistics.

Matthew S. Dryer, David Gil, Bernard Comrie, Hagen Jung, Claudia Schmidt, et al. 2005. The world atlas of language structures.

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.

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.

Shai Fine, Yoram Singer, and Naftali Tishby. 1998. The hierarchical hidden Markov model: Analysis and applications. *Machine Learning*, 32(1):41–62.

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, Valencia, Spain. Association for Computational Linguistics.

Geoffrey E. Hinton, Peter Dayan, Brendan J. Frey, and Radford M. Neal. 1995. The "wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161.

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

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.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. 1999. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.

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 (ACL)*, pages 555–560, Berlin, Germany. Association for Computational Linguistics.

Aleksandr E. Kibrik. 1998. Archi. In Andrew Spencer and Arnold M. Zwicky, editors, *The Handbook of Morphology*, pages 455–476.

Diederik P. Kingma and Max Welling. 2013. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.

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

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Thomas Müller, Ryan Cotterell, Alexander Fraser, and Hinrich Schütze. 2015. Joint lemmatization and morphological tagging with Lemming. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2268–2274, Lisbon, Portugal. 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, Seattle, Washington, USA. 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.

Joakim Nivre, Željko Agić, Lars Ahrenberg, Maria Jesus Aranzabe, Masayuki Asahara, Aitziber Atutxa, Miguel Ballesteros, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Eckhard Bick, Cristina Bosco, Gosse Bouma, Sam Bowman, Marie Candito, Gülşen Cebiroğlu Eryiğit, Giuseppe G. A. Celano, Fabricio Chalub, Jinho Choi, Çağrı Çöltekin, Miriam Connor, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, and Dobrovoljc. 2017. Universal dependencies 2.0. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics ('UFAL), Faculty of Mathematics and Physics, Charles University.

Lawrence R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Andrew Spencer. 1991. *Morphological Theory: An Introduction to Word Structure in Generative Grammar*. Wiley-Blackwell.

Andreas Stuhlmüller, Jacob Taylor, and Noah Goodman. 2013. Learning stochastic inverses. In *Advances in Neural Information Processing Systems*, pages 3048–3056.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *Thirteenth Annual Conference of the International Speech Communication Association*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

John Sylak-Glassman. 2016. The composition and use of the universal morphological feature schema (Unimorph schema). Technical report, Johns Hopkins University.

John Sylak-Glassman, Christo Kirov, David Yarowsky, and Roger Que. 2015. A language-independent feature schema for inflectional morphology. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL)*, pages 674–680, Beijing, China. Association for Computational Linguistics.

Matthew D. Zeiler. 2012. Adadelta: An adaptive learning rate method. *arXiv preprint:1212.5701*.

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.

A Hyperparameters and Experimental Details

Here, we list all the hyperparameters and other experimental details necessary for the reproduction of the numbers presented in Tab. 3. The final experiments were produced with the follow setting. We performed a modest grid search over various configurations in the search of the best option on development for each component.

LSTM Morphological Tag Language Model. The morphological tag language model is a 2-layer vanilla LSTM trained with hidden size of 200. It is trained to for 40 epochs using SGD with a cross entropy loss objective, and an initial learning rate of 20 where the learning rate is quartered during any epoch where the loss on the validation set reaches a new minimum. We regularize using dropout of 0.2 and clip gradients to 0.25. The morphological tags are embedded (both for input and output) with a multi-hot encoding into $\mathbb{R}^{200}$, where any given tag has an embedding that is the sum of the embedding for its constituent POS tag and each of its constituent slots.

Lemmata Generator. The lemma generator is a single-layer vanilla LSTM, trained for 10000 epochs using SGD with a learning rate of 4, using a batch size of 20000. The LSTM has 50 hidden units, embeds the POS tags into $\mathbb{R}^5$ and each token (i.e., character) into $\mathbb{R}^5$. We regularize using weight decay (1e-6), no dropout, and clip gradients to 1. When sampling lemmata from the model, we cool the distribution using a temperature of 0.75 to generate more "conservative" values. The hyperparameters were manually tuned on Latin data to produce sensible output and fit development data and then reused for all languages of this paper.

Morphological Inflector. The reinflection model is a single-layer GRU-cell seq2seq model with a bidirectional encoder and multiplicative attention in the style of Luong et al. (2015), which we train for 250 iterations of AdaDelta (Zeiler, 2012). Our search over the remaining hyperparameters was as follows (optimal values in bold): input embedding size of [50, 100, **200**, 300], hidden size of [50, **100**, 150, 200], and a dropout rate of [0.0, 0.1, 0.2, 0.3, 0.4, **0.5**].

Lemmatizer and Morphological Tagger. The joint lemmatizer and tagger is LEMMING as described in §5.5. It is trained with default parameters, the pretrained word vectors from Bojanowski et al. (2016) as type embeddings, and beam size 3.

Wake-Sleep We run two iterations ($I = 2$) of wake-sleep. Note that each of the subparts of wake-sleep: estimating $p_{\boldsymbol{\theta}}$ and estimating $q_{\boldsymbol{\phi}}$ are trained to convergence and use the hyperparameters described in the previous paragraphs. We set γ_{wake} and γ_{sleep} to 0.25, so we observe roughly $1/4$ as many dreamt samples as true samples. The samples from the generative model often act as a regularizer, helping the variational approximation (as measured on morphological tagging and lemmatization accuracy) on the UD development set, but sometimes the noise lowers performance a mite. Due to a lack of space in the initial paper, we did not deeply examine the performance of the tagger-lemmatizer outside the context of improving inflection prediction accuracy. Future work will investigate question of how much tagging and lemmatization can be improved through the incorporation of samples from our generative model. In short, our efforts will evaluate the inference network in its own right, rather than just as a variational approximation to the posterior.

B Fake Data from the Sleep Phase

An example sentence $\tilde{\boldsymbol{f}}$ sampled via $\langle \tilde{\boldsymbol{f}}, \tilde{\boldsymbol{\ell}}, \tilde{\boldsymbol{m}} \rangle \sim p_{\boldsymbol{\theta}}(\cdot, \cdot, \cdot)$ in Portuguese:

```
dentremeticamente » isso Procusas
Da Fase » pos a acordítica
Máisringeringe Ditudis A ana ,
Urevirao Da De O linsith.muital ,
E que chegou interalionalmente Da
anundica De mêpinsuriormentais .
```

and in Latin:

```
inpremcret ita sacrum super annum
pronditi avocere quo det tuam
nunsidebus quod puela ?
```