Neural Graphical Models over Strings for Principal Parts Morphological Paradigm Completion

Ryan Cotterell[1,2] **John Sylak-Glassman**[2] **Christo Kirov**[2]
[1]Department of Computer Science
[2]Center for Language and Speech Processing
Johns Hopkins University
ryan.cotterell@jhu.edu jcsg@jhu.edu ckirov@gmail.com

Abstract

Many of the world's languages contain an abundance of inflected forms for each lexeme. A major task in processing such languages is predicting these inflected forms. We develop a novel statistical model for the problem, drawing on graphical modeling techniques and recent advances in deep learning. We derive a Metropolis-Hastings algorithm to jointly decode the model. Our Bayesian network draws inspiration from principal parts morphological analysis. We demonstrate improvements on 5 languages.

1 Introduction

Inflectional morphology modifies the form of words to convey grammatical distinctions (e.g. tense, case, and number), and is an extremely common and productive phenomenon throughout the world's languages (Dryer and Haspelmath, 2013). For instance, the Spanish verb *poner* may transform into one of over fifty unique inflectional forms depending on context, e.g. the 1st person present form is *pongo*, but the 2nd person present form is *pones*. These variants cause data sparsity, which is problematic for machine learning since many word forms will not occur in training corpora. Thus, a necessity for improving NLP on morphologically rich languages is the ability to analyze all inflected forms for any lexical entry. One way to do this is through paradigm completion, which generates all the inflected forms associated with a given lemma.

Until recently, paradigm completion has been narrowly construed as the task of generating a full paradigm (e.g. noun declension, verb conjugation) based on a single privileged form—the lemma (i.e. the citation form, such as *poner*). While recent work (Durrett and DeNero, 2013; Hulden, 2014; Nicolai et al., 2015; Ahlberg et al., 2015; Faruqui et al., 2016) has made tremendous progress on this narrower task, paradigm completion is not only broader in scope, but is better solved without privileging the lemma over other forms. By forcing string-to-string transformations from one inflected form to another to go through the lemma, the transformation problem is often made more complex than by allowing transformations to happen directly or through a different intermediary form. This interpretation is inspired by ideas from linguistics and language pedagogy, namely principal parts morphology, which argues that forms in a paradigm are best derived using a *set* of citation forms rather than a single form (Finkel and Stump, 2007a; Finkel and Stump, 2007b).

Directed graphical models provide a natural formalism for principal parts morphology since a graph topology can represent relations between inflected forms and principal parts. Specifically, we apply string-valued graphical models (Dreyer and Eisner, 2009; Cotterell et al., 2015) to the problem. We develop a novel, neural parameterization of string-valued graphical models where the conditional probabilities in the Bayesian network are given by a sequence-to-sequence model (Sutskever et al., 2014). However, under such a parameterization, exact inference and decoding are intractable. Thus, we derive a sampling-based decoding algorithm. We experiment on 5 languages: Arabic, German, Latin, Russian, and Spanish, showing that our model outperforms a baseline approach that privileges the lemma form.

2 A Generative Model of Principal Parts

We first formally define the task of paradigm completion and relate it to research in principal parts morphology. Let Σ be a discrete alphabet of characters in a language. Formally, for a given lemma

Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pages 759–765,
Valencia, Spain, April 3-7, 2017. ©2017 Association for Computational Linguistics

$\ell \in \Sigma^*$, we define the complete paradigm of that lemma $\pi(\ell) = \langle m_1, \ldots, m_N \rangle$, where each $m_i \in \Sigma^*$ is an inflected form.[1] For example, the paradigm for the English lemma *to run* is defined as $\pi\,(\text{RUN}) = \langle \textit{run}, \textit{runs}, \textit{ran}, \textit{running} \rangle$. While the size of a typical English verbal paradigm is comparatively small ($|\pi| = N = 4$), in many languages the size of the paradigms can be very large (Kibrik, 1998). The task of paradigm completion is to predict all elements of the tuple π given one or more forms (m_i). Paradigm completion solely from the lemma (m_ℓ), however, largely ignores the linguistic structure of the paradigm. Given certain inflected word forms, termed principal parts, the construction of a set of other word forms in the same paradigm is fully deterministic. Latin verbs are famous for having four such principal parts (Finkel and Stump, 2009).

Inspired by the concept of principal parts, we present a solution to the paradigm completion task in which target inflected forms are predicted from other forms in the paradigm, rather than only from the lemma. We implement this solution in the form of a generative probabilistic model of the paradigm. We define a joint probability distribution over the entire paradigm:

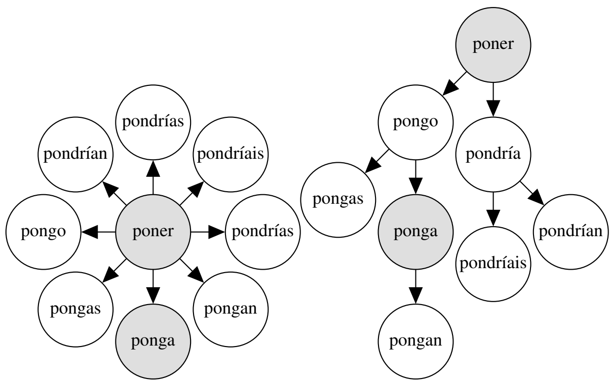
$$p(\pi) = \prod_i p(m_i \mid m_{\text{pa}_{\mathcal{T}}(i)}) \tag{1}$$

where $\text{pa}_{\mathcal{T}}(\cdot)$ is a function that returns the parent of the node i with respect to the tree $\mathcal{T}$, which encodes the source form from which each target form is predicted. In terms of graphical modeling, this $p(\pi)$ is a Bayesian network over string-valued variables (Cotterell et al., 2015). Trees provide a natural formalism for encoding the intuition behind principal parts theory, and provide a fixed paradigm structure prior to inference. We construct a graph with nodes for each cell in the paradigm, as in Figure 1. The parent of each node is another form in the paradigm that best predicts that node.

2.1 Paradigm Trees

Baseline Network. Predicting inflected forms only from the lemma involves a particular graphical model in which all the forms are leaves attached to the lemma. This network is treated as a baseline, and is depicted in Figure 1a.

[1] We constrain the task such that the number of forms in a paradigm ($|\pi| = N$) is fixed, and each possible form of a paradigm is assumed to have consistent semantics.



(a) Lemma paradigm tree. (b) Principal parts paradigm tree.

Figure 1: Two potential graphical models for the paradigm completion task. The topology in (a) encodes the the network where all forms are predicted from the lemma. The topology in (b) is a principle-parts-inspired topology introduced here.

Heuristic Network. We heuristically induce a paradigm tree with the following procedure. For each ordered pair of forms in a paradigm π, we compute the number of distinct edit scripts that convert one form into the other. The edit script procedure is similar to that described in Chrupała et al. (2008). For each ordered pair (i, j) of inflected forms in π, we count the number of distinct edit paths mapping from m_i to m_j, which serves as a weight on the edge $w_{i \to j}$. Empirically, $w_{i \to j}$ is a good proxy for how deterministic a mapping is. We use Edmonds' algorithm (Edmonds, 1967) to find the minimal directed spanning tree. The intuition behind this procedure is that the number of deterministic arcs should be maximized.

Gold Network. Finally, for Latin verbs we consider a graph that matches the classic pedagogical derivation of Latin verbs from four principal parts.

3 Inflection Generation with RNNs

RNNs have recently achieved state-of-the-art results for many sequence-to-sequence mapping problems and paradigm completion is no exception. Given the success of LSTM-based (Hochreiter and Schmidhuber, 1997) and GRU-based (Cho et al., 2014) morphological inflectors (Faruqui et al., 2016; Cotterell et al., 2016), we choose a neural parameterization for our Bayesian network, i.e. the conditional probability $p(m_i \mid m_{\text{pa}_{\mathcal{T}}(i)})$ is computed using a RNN. Our graphical modeling approach as well as the inference algorithms subsequently discussed in §4.2 are agnostic to the minutiae of any one parameterization, i.e. the encoding $p(m_i \mid m_{\text{pa}_{\mathcal{T}}(i)})$ is a black box.

Algorithm 1 Decoding by Simulated Annealing

procedure SIMULATED-ANNEALING($\mathcal{T}$, d, ϵ)
  $\tau \leftarrow 10.0$; $\boldsymbol{m} \leftarrow [\varepsilon, \ldots, \varepsilon]$
  repeat
    $i \sim \text{uniform}(\{1, 2, \ldots, |\boldsymbol{m}|\})$ ▷ sample latent node in $\mathcal{T}$
    $m'_i \sim q_i$ ▷ sample string from proposal distribution
    $a \leftarrow \min\left[1, \left(\frac{p(m'_i \mid m_{\text{pa}_{\mathcal{T}}(i)})}{p(m_i \mid m_{\text{pa}_{\mathcal{T}}(i)})}\right)^{1/\tau} \frac{q_i(m_i)}{q_i(m'_i)}\right]$
    if uniform$(0, 1) \leq a$ **then**
      $m_i \leftarrow m'_i$ ▷ update string to new value if accepted
    $\tau \leftarrow \tau \cdot d$ ▷ decay temperature where $d \in (0, 1)$
  until $\tau \leq \epsilon$ ▷ repeat until convergence; see Spall (2003, Ch. 8)
  return $\boldsymbol{m}$

3.1 LSTMs with Hard Monotonic Attention

We define the conditional distributions in our Bayesian network $p(m_i \mid m_{\text{pa}_{\mathcal{T}}(i)})$ as LSTMs with hard monotonic attention (Aharoni et al., 2016; Aharoni and Goldberg, 2016), which we briefly overview. These networks map one inflection to another, e.g. mapping the English gerund *running* to the past tense *ran*, using an encoder-decoder architecture (Sutskever et al., 2014) run over an augmented alignment alphabet, consisting of copy, substitution, deletion and insertion, as in Dreyer et al. (2008). For strings $x, y \in \Sigma^*$, the alignment is extracted from the minimal weight edit path using the BioPython toolkit (Cock et al., 2009). Crucially, as the model is locally normalized we may sample strings from the conditional $p(m_i \mid m_{\text{pa}_{\mathcal{T}}(i)})$ *efficiently* using forward sampling. This network stands in contrast to attention models (Bahdanau et al., 2015) in which the alignments are soft and not necessarily monotonic. We refer the reader to Aharoni et al. (2016) for exact implementation details as we use their code out-of-the-box.[2]

4 Neural Graphical Models over Strings

Our Bayesian network defined in Equation (1) is a graphical model defined over multiple string-valued random variables, a framework formalized in Dreyer and Eisner (2009). In contrast to previous work, e.g. Cotterell and Eisner (2015; Peng et al. (2015), which considered conditional distributions encodable by finite-state machines, we offer the first neural parameterization for such graphical models. With the increased expressivity comes computational challenges—inference becomes intractable. Thus, we fashion an efficient sampling algorithm.

[2] `https://github.com/roeeaharoni/morphological-reinflection`

4.1 Parameter Estimation

Following previous work (Faruqui et al., 2016), we train our model in the fully observed setting with *complete paradigms* as training data. As our model is directed, this makes parameter estimation relatively straightforward. We may estimate the parameters of each LSTM independently without performing joint inference during training. We follow the training procedure of Aharoni et al. (2016), using a maximum of 300 epochs of SGD.

4.2 Approximate Joint Decoding

In a Bayesian network, the *maximum-a-posteriori* (MAP) inference problem refers to finding the most probable configuration of the variables given some evidence. In our case, this requires finding the best set of inflections to complete the paradigm given an observed set of inflected forms. Returning to the English verbal paradigm, given the past tense form *ran* and the 3rd person present singular *runs*, the goal of MAP inference is to return the most probable assignment to the past tense and gerund form (the correct assignment is *ran* and *running*). In many Bayesian networks, e.g. models with finite support, exact MAP inference can be performed efficiently with the sum-product belief propagation algorithm (Pearl, 1988) when the model has a tree structure. Despite the tree structure, the LSTM makes exact inference intractable. Thus, we resort to an approximate scheme.

4.3 Simulated Annealing

The Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970) is a popular Markov-Chain Monte Carlo (MCMC) (Robert and Casella, 2013) algorithm for approximate sampling from intractable distributions. As with all MCMC algorithms, the goal is to construct a Markov chain whose stationary distribution is the target distribution. Thus, after having mixed, taking a random walk on the Markov chain is equivalent to sampling from the intractable distribution. Here, we are interested in sampling from $p(\pi)$, where part of π may be observed.

Simulated annealing (Kirkpatrick et al., 1983; Andrieu et al., 2003) is a slight modification of the Metropolis-Hastings algorithm suitable for MAP inference. We add the temperature hyperparameter τ, which we decrease on a schedule. We achieve the MAP estimate as $\tau \rightarrow 0$. The algorithm works as follows: Given a paradigm with tree $\mathcal{T}$, we sam-

Language	Baseline	Heuristic Tree	Gold Tree
Arabic	70.3%	92.7%	N/A
German	93.3%	98.8%	N/A
Latin	92.8%	98.3%	98.9%
Russian	84.2%	84.4%	N/A
Spanish	99.2%	99.2%	N/A

Table 1: Accuracy on the paradigm completion task comparing Bayesian network topologies over 5 languages.

Language (POS)	Train	Dev	Test
Arabic (N)	632	79	79
German (N)	1723	200	200
Latin (V)	2660	333	333
Russian (N)	8266	1032	1033
Spanish (V)	2973	372	372

Table 2: Lemmata per dataset.

ple a latent node i in the tree uniformly at random. We then sample a new string m'_i from the proposal distribution q_i (see §4.4), which we accept (replacing m_i) with probability

$$a{=}\min\left[1, \left(\frac{p(m'_i \mid m_{\mathrm{pa}_{\mathcal{T}}(i)})}{p(m_i \mid m_{\mathrm{pa}_{\mathcal{T}}(i)})}\right)^{1/\tau} \frac{q_i(m_i)}{q_i(m'_i)}\right]. \quad (2)$$

We iterate until convergence and accept the final configuration of values as our approximate MAP estimate. We give pseudocode in Algorithm 1 for clarity.

4.4 Proposal Distribution

We define a tractable proposal distribution for our neural graphical model over strings using a procedure similar to the stochastic inverse method of Stuhlmüller et al. (2013) for probabilistic programming. In addition to estimating the parameters of an LSTM defining the distribution $p(m_i \mid m_{\mathrm{pa}_{\mathcal{T}}(i)})$, we also estimate parameters of an LSTM to define the *inverse distribution* $p(m_{\mathrm{pa}_{\mathcal{T}}(i)} \mid m_i)$. As we observe only complete paradigms at training time, we train networks as in §4.1. First, we define the neighborhood of a node i as all those nodes adjacent to i (connected by an ingoing *or* outgoing edge). We define the proposal distribution as a mixture model of all conditional distributions in the neighborhood $\mathcal{N}(i)$, i.e.

$$q_i(m_i) = |\mathcal{N}(i)|^{-1} \sum_{j \in \mathcal{N}(i)} p(m_i \mid m_j). \quad (3)$$

Crucially, some of the distributions are stochastic inverses. Sampling from q_i is tractable: We sample a mixture component uniformly and then sample a string.

5 Related Work

Our effort is closest to Faruqui et al. (2016), who proposed the first neural paradigm completer. Many neural solutions were also proposed in the SIGMORPHON shared task on morphological reinflection (Cotterell et al., 2016). Notably, the winning system used an encoder-decoder architecture (Kann and Schütze, 2016). Neural networks have been used in other areas of computational morphology, e.g. morpheme segmentation (Wang et al., 2016; Kann et al., 2016; Cotterell and Schütze, 2017), morphological tagging (Heigold et al., 2016), and language modeling (Botha and Blunsom, 2014).

6 Experiments and Results

Our proposed model *generalizes* previous efforts in paradigm completion since all previously proposed models take the form of Figure 1a, i.e. a graphical model where all leaves connect to the lemma. Unfortunately, in that configuration, observing additional forms *cannot* help at test time since information must flow through the lemma, which is always observed. We conjecture that principal parts-based topologies will outperform the baseline topology for that reason. We propose a controlled experiment in which we consider identical training and testing conditions and vary only the topology.

Data. Data for training, development, and testing is randomly sampled from the UniMorph dataset (Sylak-Glassman et al., 2015).[3] We run experiments on Arabic, German, and Russian nominal paradigms and Latin and Spanish verbal paradigms. The sizes of the resulting data splits are given in Table 2. For the development and test splits we always include the lemma (as is standard) while sampling additional observed forms. On average one third of all forms are observed.

Evaluation. Evaluation of the held-out sets proceeds as follows: Given the observed forms in the paradigm, we jointly decode the remaining forms as discussed in §4.2; joint decoding is performed without Algorithm 1 for the baseline—instead, we

[3] http://www.unimorph.org

decode as in Aharoni et al. (2016). We measure accuracy (macro-averaged) on the held-out forms.

Results. In general, we find that our principal parts-inspired networks outperform lemma-centered baseline networks. In Arabic, German, and Latin, we find the largest gains (for Latin, our heuristic topology closely matches that of the gold tree, validating the heuristics we use). We attribute the gains to the ability to use knowledge from attested forms that are otherwise difficult to predict, e.g. forms based on the Arabic broken plural, the German plural, and any of the Latin present perfect forms. In the case of paradigms with portions which are difficult to predict without knowledge of a representative form, knowing multiple principle parts will be a boon given a proper tree improvement—we attribute this to the fact that almost all of the test examples were regular *-ar* verbs and, thus, fully predictable. Finally, in the case of Russian we see only minor improvements—this stems from need to maintain a different optimal topology for each declension. Because our model assumes a fixed paradigmatic structure in the form of a tree, using multiple topologies is not possible.

7 Conclusion

We have presented a directed graphical model over strings with a RNN parameterization for principle-parts-inspired morphological paradigm completion. This paradigm gives us the best of two worlds. We can exploit state-of-the-art neural morphological inflectors while injecting linguistic insight into the structure of the graphical model itself. Due to the expressivity of our parameterization, exact decoding becomes intractable. To solve this, we derive an efficient MCMC approach to approximately decode the model. We validate our model experimentally and show gains over a baseline which represents the topology used in nearly all previous research.

Acknowledgements

The first author was supported by a DAAD Long-Term Research Grant and an NDSEG fellowship.

References

Roee Aharoni and Yoav Goldberg. 2016. Sequence to sequence transduction with hard monotonic attention. *arXiv preprint arXiv:1611.01487*.

Roee Aharoni, Yoav Goldberg, and Yonatan Belinkov. 2016. Improving sequence to sequence learning for morphological inflection generation: The biu-mit systems for the sigmorphon 2016 shared task for morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 41–48, Berlin, Germany, August. 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, May–June. Association for Computational Linguistics.

Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I. Jordan. 2003. An introduction to MCMC for machine learning. *Machine Learning*, 50(1-2):5–43.

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

Jan A. Botha and Phil Blunsom. 2014. Compositional morphology for word representations and language modelling. In *ICML*, pages 1899–1907.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, pages 1724–1734.

Grzegorz Chrupała, Georgiana Dinu, and Josef van Genabith. 2008. Learning morphology with Morfette. In *LREC*, volume 8.

Peter Cock, Tiago Antao, Jeffrey Chang, Brad Chapman, Cymon Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, et al. 2009. Biopython: Freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423.

Ryan Cotterell and Jason Eisner. 2015. Penalized expectation propagation for graphical models over strings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 932–942, Denver, Colorado, May–June. Association for Computational Linguistics.

Ryan Cotterell and Hinrich Schütze. 2017. Joint semantic synthesis and morphological analysis of the derived word. *CoRR*, abs/1701.00946.

Ryan Cotterell, Nanyun Peng, and Jason Eisner. 2015. Modeling word forms using latent underlying morphs and phonology. *Transactions of the Association for Computational Linguistics*, 3.

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

Markus Dreyer and Jason Eisner. 2009. Graphical models over multiple strings. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 101–110, Singapore, August. Association for Computational Linguistics.

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*, pages 1080–1089, Honolulu, Hawaii, October. Association for Computational Linguistics.

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

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, June. Association for Computational Linguistics.

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

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

Raphael Finkel and Gregory Stump. 2007a. Principal parts and degrees of paradigmatic transparency (no. tr 470-07). Technical report, Department of Computer Science, University of Kentucky, Lexington, KY.

Raphael Finkel and Gregory Stump. 2007b. Principal parts and morphological typology. *Morphology*, 17:39–75, November.

Raphael Finkel and Gregory Stump. 2009. What your teacher told you is true: Latin verbs have four principal parts. *Digital Humanities Quarterly (DHQ)*, 3(1).

Wilfred Keith Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Georg Heigold, Guenter Neumann, and Josef van Genabith. 2016. Neural morphological tagging from characters for morphologically rich languages. *CoRR*, abs/1606.06640.

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

Mans Hulden. 2014. Generalizing inflection tables into paradigms with finite-state operations. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 29–36, Baltimore, Maryland. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2016. MED: The LMU system for the SIGMORPHON 2016 shared task on morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 62–70, Berlin, Germany, August. Association for Computational Linguistics.

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2016. Neural morphological analysis: Encoding-decoding canonical segments. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 961–967, Austin, Texas, November. Association for Computational Linguistics.

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

Scott Kirkpatrick, C. Daniel Gelatt, Mario P. Vecchi, et al. 1983. Optimization by simmulated annealing. *Science*, 220(4598):671–680.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 1953. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092.

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, May–June. Association for Computational Linguistics.

Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

Nanyun Peng, Ryan Cotterell, and Jason Eisner. 2015. Dual decomposition inference for graphical models over strings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 917–927, Lisbon, Portugal, September. Association for Computational Linguistics.

Christian Robert and George Casella. 2013. *Monte Carlo statistical methods*. Springer Science & Business Media.

James C. Spall. 2003. *Introduction to stochastic search and optimization: estimation, simulation, and control*, volume 65. John Wiley & Sons.

Andreas Stuhlmüller, Jacob Taylor, and Noah Goodman. 2013. Learning stochastic inverses. In *NIPS*, pages 3048–3056.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215.

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 (Volume 2: Short Papers)*, pages 674–680, Beijing, China, July. Association for Computational Linguistics.

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