The OSU Realizer for SRST '18: Neural Sequence-to-Sequence Inflection and Incremental Locality-Based Linearization

David L. King
The Ohio State University
Department of Linguistics
Columbus, Ohio
king.2138@osu.edu

Michael White
The Ohio State University
Department of Linguistics
Columbus, Ohio
mwhite@ling.osu.edu

Abstract

Surface realization is a nontrivial task as it involves taking structured data and producing grammatically and semantically correct utterances. Many competing grammar-based and statistical models for realization still struggle with relatively simple sentences. For our submission to the 2018 Surface Realization Shared Task, we tackle the shallow task by first generating inflected wordforms with a neural sequence-to-sequence model before incrementally linearizing them. For linearization, we use a global linear model trained using early update that makes use of features that take into account the dependency structure and dependency locality. Using this pipeline sufficed to produce surprisingly strong results in the shared task. In future work, we intend to pursue joint approaches to linearization and morphological inflection and incorporating a neural language model into the linearization choices.

1 Introduction

We participated in the surface track of the 2018 Surface Realization Shared Task (Mille et al., 2018, SRST '18). In the surface track, task inputs were created by extracting sentences in 10 languages from the Universal Dependency treebanks corpus, scrambling the words and converting them to their citation form. The task was then to generate a natural and semantically adequate sentence by inflecting and ordering the words.

Our aims in participating in the shared task were twofold. First, we aimed to investigate the extent to which neural sequence-to-sequence models developed for the 2016 and 2017 SIGMORPHON shared tasks on morphological reinflection (Faruqui et al., 2016; Kann and Schütze, 2016) could be adapted to the more realistic setting for generation of SRST '18. Second, we aimed to investigate the extent to which dependency locality (Gibson, 2000) features previously shown to be important for grammar-based generation in English (White and Rajkumar, 2012) and in corpus-based studies of syntactic choice (Temperley, 2007; Liu, 2008; Gildea and Temperley, 2010; Rajkumar et al., 2016) would also prove effective with incremental, dependency-based linearization (Liu et al., 2015; Puduppully et al., 2016) across languages.

At an overview level, our system treats the task of surface realization as a simple two-stage process. First, we convert uninflected lexemes to fully inflected wordforms using the grammatical features supplied by the UD corpus; and second, we incrementally linearize the inflected words using the supplied syntactic dependencies, grammatical features and locality-based features that take dependency length and phrase size into account. A simple rule-based detokenizer attaches punctuation to adjacent words in a final step. The system was trained using only the supplied data. We leave for future work investigating ways to jointly make inflection and linearization choices and to incorporate a neural language model.

2 Background

The intuition behind using neural and statistical models for learning morphology originated with what Ackerman et al. (2009) referred to as the *Paradigm Cell Filling Problem* (PCFP). For any given learner, human or machine, there exists no input such that exposing the learner to that input will also expose the learner to every possible inflected wordform. Nevertheless, humans can rou-

Person	Singular	Plural
1st	ich singe	???
2nd	du singst	???
3rd	???	sie singen

Person	Singular	Plural
1st	???	wir hören
2nd	???	ihr hört
3rd	er/sie/es hört	???

Table 1: For SRST '18, our hypothesis is that our system will not see every fully inflected word form in the training data. For example, given partial paradigms for the German verbs for SINGEN ('to sing') and HÖREN ('to hear'), we should have enough information for our system to learn the paradigm of TRINKEN, given only its citation form.

tinely and accurately fill in paradigm tables for wordforms they may have never even produced before. For any language learner, the PCFP states that a learner must take incomplete input (as seen in Table 1) and be able to produce fully inflected paradigm tables for novel words (e.g. Table 2).

Person	Singular	Plural
1st	ich trinke	wir trinken
2nd	du trinkst	ihr trinkt
3rd	er/sie/es trinkt	sie trinken

Table 2: The inferred paradigm for TRINKEN ('to drink') as learned by the partial paradigms for SINGEN ('to sing') and HÖREN ('to hear') from Table 1.

There has been extensive work computationally to combat the PCFP (Nicolai et al., 2015; Durrett and DeNero, 2013) and multiple shared tasks (Cotterell et al., 2016, 2017). Recently, models utilizing recurrent neural networks have proven most effective at the PCFP and have produced state-of-the-art results in the last two SIGMORPHON shared tasks (Kann and Schütze, 2016). Although we think this approach lends itself to our task, in that we need to produce fully inflected wordforms along with linearizing them, Kann and Schütze's system has only been tested on SIGMORPHON data and, to our knowledge, has never been used in a downstream task such as surface realization.

Turning now to dependency locality, Rajkumar et al. (2016) provide an overview of the literature on how locality considerations affect syntactic choice in human language production. The tendency to minimize dependency length has a long history of study going back to Behaghel's (1932) principle of end weight. More recently, Hawkins (1994; 2004) and Gibson (2000) have advanced theories contending that ease of production and comprehension favors a preference for dependency locality, bolstered not only by the corpus studies cited earlier but also a wide range of experimental studies. Rajkumar et al. (2016) additionally demonstrate a significant preference for dependency locality in syntactic choice even in the presence of strong controls for surprisal and memory depth. In Section 6, we show that the features we designed to capture locality preferences yield impressive gains on automatic metric scores across languages in the context of our incremental linearization system.

3 Morphological Inflection

As an initial stage in our realization process, we first predict the fully inflected wordforms from the supplied lexemes. We inflect the morphological forms before linearization in order to allow the surface forms to be used as features for linearization, but acknowledge that these steps would ideally be done jointly. For English, a high resource language, morphological inflection is relatively simple to do with existing rule-based resources like MorphG.[1] To predict fully inflected word forms in other languages as well, we exploit recent advances in neural machine translation (NMT) as implemented by Kann and Schütze in the two most recent SIGMORPHON shared tasks. Their system is based on Bahdanau et al.'s (2014) attention-based NMT architecture and models the task of wordform prediction as a kind of *translation* of one sequence to another.

Figure 1 shows the original architecture developed by Faruqui et al. (2016). Given Kann and Schütze's success in adapting this architecture to work with the SIGMORPHON data, we adopt their architecture hypothesizing that it will generalize to the SRST '18 data. The architecture uses gated recurrent units (Chung et al., 2015, GRU), a kind of recurrent neural network, whose hidden state h_t depends on the current input x_t, the previous hidden state h_{t-1}, and nonlinear function f

[1] https://github.com/knowitall/morpha

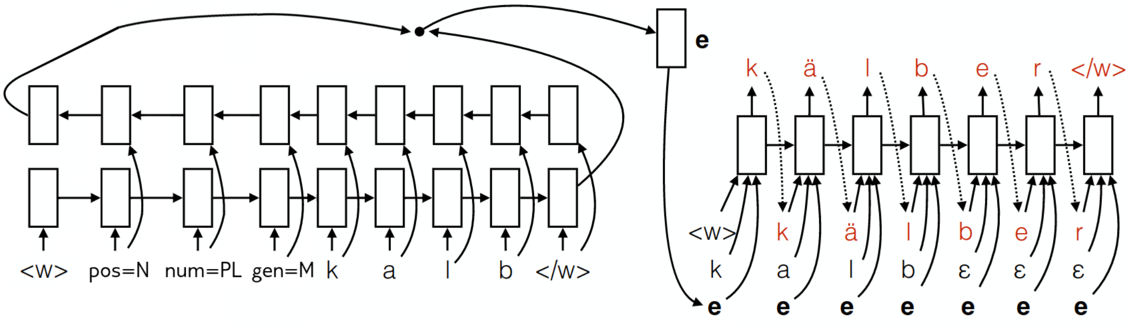

Figure 1: A graphical representation of the architecture originally introduced by Faruqui et al. (2016) and adapted by Kann and Schütze (2016) to handle SIGMORPHON2016 input. A bidirectional GRU creates an encoding of the input wordform and supplied features. That encoding is subsequently fed to the decoder GRU along with the original input wordform.

at time t. Similarly, context c for a given sequence is defined as the output from nonlinear function q over all the hidden states from time step 1 to t over the length of sequence x.

$$h_t = f(x_t, h_{t-1}) \tag{1}$$

$$c = q(h_1, ..., h_{T_x}) \tag{2}$$

Since we used a bidirectional GRU, we set h_j to be the the concatenated vectors of the forwards and backwards encoding of the sequence:

$$h_j = \left[\overrightarrow{h_j^T}, \overleftarrow{h_j^T}\right]^T \tag{3}$$

We define inference (the decoding step) of output y given input sequence x as a distribution of possible output strings:

$$p(y|x) = \prod_{t=1}^{T_y} p(y_t|\{y_1, ..., y_t - 1\}, s_t, c_t) \tag{4}$$

This distribution is derived from the product of previous individual outputs y_1, y_2, ..., y_{t-1} up to the current time step t to produce the most likely output y_t. Output y is also dependent on s_t (the hidden state of the decoder) and context c_t (the weighted sum of annotations produced by the encoder):

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{5}$$

Where we calculate weights α_{ij} for h_j as:

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T_x} exp(e_{ik})} \tag{6}$$

$$e_{ij} = a(s_{i-1}, hj) \tag{7}$$

We used standard cross-entropy loss, 300 hidden units for both the encoder and decoder. We followed Kann and Schütze by training the model using minibatches of 20 and Adadelta (Zeiler, 2012). For the datasets, we used the entirety of the supplied training data, but only used a random sample of 6000 items from the development set to speed up training. Models for each language were trained until wordform prediction accuracy on the development set was over 98% or up to 30 epochs with early stopping. Dropout was set to 0.5.

Table 3 shows our system's performance in selecting fully inflected wordforms on the development set. We also supply two competing baselines as a point of comparison: one in which our system just copies the citation form supplied and one where it only selects the most common inflected wordform seen in training. By and large, we see tremendous improvements in selecting the correct wordform.

Our final feature set included any features supplied by the data, in addition to features from immediate children and parents in the dependency tree. We made use of all features from a given

	Language									
Model	ar	cs	en	es	fi	fr	it	nl	pt	ru
Lemma	11.5	32.9	67.0	45.5	24.9	48.9	50.8	62.9	51.5	6.2
Majority	50.7	43.0	67.5	59.2	26.2	58.5	58.9	65.1	60.3	34.1
MED	92.3	91.7	89.2	99.2	98.6	98.3	92.1	88.5	96.1	90.1

Table 3: Morphological inflection results on the development set compared to baseline results of simply copying the lemma or using the most frequent inflected wordform.

word and any features from any parent word. We also chose to exclusively add features from children with argument relations (i.e. *dobj*, *nsubj*, *csubj*, etc.), with the intuition that, for example, the argument of a verb would influence a given verb's inflection, while an adverb might not. To illustrate this, as seen in Figure 2, in the fragment DIT IS MOOI ... ('That is beautiful'), the features from DIT facilitate properly inflecting the verb ZIJN ('to be') as IS ('is') and not BENT or BEN ('are' or 'am' respectively), since the feature 'Person=3' is not encoded in the copula, but rather in the pronoun. Meanwhile the *advmod* relation is not helpful in informing our system how to inflect ZIJN.

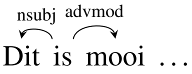

Figure 2: An example (from the Dutch training set) of how child dependencies with argument relations help with inflection, while other modifier relations do not. The person features in DIT help to realize ZIJN as IS and not BENT or BEN. However, the features from the *advmod* relation are not helpful.

4 Linearization

Previous work on dependency-based surface realization (Bangalore and Rambow, 2000; Filippova and Strube, 2007, 2009; Guo et al., 2008; Bohnet et al., 2010, 2011; Guo et al., 2011; Zhang and Clark, 2015) has emphasized bottom-up approaches that make relatively little use of dependency locality. For this task, we opted to follow Liu et al. (2015); Puduppully et al. (2016) in taking an incremental approach to linearization so as to be compatible with future work incorporating neural language models (Wen et al., 2015; Dušek and Jurcicek, 2016; Konstas et al., 2017) while giving greater emphasis to locality considerations.

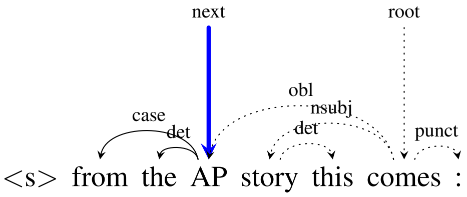

Figure 3: Example candidate realization, with *AP* as the next word and remaining words (with dotted dependencies) still in the randomized input order.

In our approach, a candidate realization is a (partial) permutation of the input words. Candidates are generated by extending a previous candidate with an input word that has not yet been chosen, as illustrated in Figure 3. Since the number of candidates is factorial in the number of input words, beam search is employed with scores computed using a global linear model. By tracking the way in which the input words are permuted, features can be calculated both from the candidate sequence as well as from the input dependency tree.

To further constrain linearization choices, projective outputs can be enforced by ensuring that all output phrases are continuous. To do so, we calculate the successors of the previous word and require the next word choice to be a descendant of one of the successors. If the previous word has child words in the dependency tree that have not yet been linearized, then the successors are the as-yet-unselected children. Otherwise, the successors include the unrealized parent and siblings of the previous word; if all those words have been covered, successors are calculated by recursing up the tree.

Since we found that 2.5% of the English development trees contained non-projective trees (even ignoring punctuation as a source of discontinuity), we opted to allow non-projective outputs to be generated. To do so, we used a discontinuity feature to encourage the model to learn that most choices should yield continuous phrases,

Events	Base Predictors	Locality Predictors
next word	trigram word, stem and POS	
dependency ordering	whether initial or final; parent and child stems, POSs, grammatical features and dep relation; sibling stem, POS and dep rel	difference in log binned size of sibling subtree
completed arc	whether projective; parent and child stems, POSs and dep relation	log binned dependency length
discontinuity	trigram POS; bigram dep relation; relation of extraposed dep	log binned size of extraposed subtree

Table 4: Linearization features, which combine events with different base and locality predictors.

where the next word is taken to introduce a discontinuity if it is not a descendant of the previous word's successors. A benefit of this soft approach to enforcing projectivity is that all sequences can be generated in principle. Note that our approach to calculating successors is similar to (though simpler than) the aforementioned transition-based approaches while also allowing limited non-projectivity.

We used scikit-learn's implementation[2] of the passive-aggressive classifier (Crammer et al., 2006) for our global linear model. The model was trained discriminatively using early update with the additional requirement that the gold candidate be top-ranked in the beam (Puduppully et al., 2016). Mini-batches were processed in parallel by averaging the updated models. To encourage faster training, we averaged the models after each mini-epoch of only a few mini-batches, rather than waiting to the end of an entire training epoch. Nevertheless, we suspect that the models were undertrained as training often failed to reach the end of longer sentences even after 12 hours of training using between 12 and 28 processors (not all languages were given 28 processors to obtain faster throughput). Looking at the training curve for English, we obtained good performance after 10 epochs but the BLEU score on the development set was still generally increasing when training timed out at 30 epochs.

Our feature set is summarized in Table 4. Features are based on four kinds of events that are calculated as each word is added: next-word events, dependency-ordering events, completed-arc events and discontinuity events. A variety of predictors are extracted for each kind of event, as shown in the table. Base features include trigram word-, POS- and stem-sequences, parent and child stems, dependency relations, parent and child grammatical features, and whether a word is initial or final in its phrase. Locality-based features additionally include binned dependency length for completed arcs, binned difference in size of sibling subtrees and binned size of any extraposed dependent subtree. For lookahead, ancestor ordering features are calculated in the same way as the head-dependent ordering features, so for example starting a sentence with a determiner entails that its head noun will precede its parent (e.g. the main verb). Features are count-based and constructed by combining each event with each of its predictors and each pair of two predictors. For example, when adding *AP* in Figure 3 as the next word, one of the constructed features pairs the previous two POS tags with the next-word event as `next_word=AP:prev2_pos=IN:prev_pos=DT` and its count is incremented.

For ease of implementation, we limited the model to count-based features. By contrast, in previous work with a bottom-up, chart-based realizer, White and Rajkumar (2012) found it helpful to include a feature whose value was the total dependency length of a constituent. In the incremental setting here, we expect that the binned relative size of siblings is the most helpful locality-based feature for ordering, as the binned dependency length feature likely does not become available in a timely fashion in the beam search with long dependencies. In future work, we plan to investigate ways to better model the total dependency length incrementally by accumulating the sum of open dependencies in candidate realizations.

[2] http://scikit-learn.org

	Language									
BLEU	ar	cs	en	es	fi	fr	it	nl	pt	ru
NoLoc	27.5	47.5	56.8	57.9	33.1	34.6	36.7	23.5	44.1	45.8
Dev	29.3	53.7	68.9	65.0	35.9	38.4	42.3	26.2	48.0	56.6
Test	25.6	53.2	66.3	65.3	37.5	38.2	42.1	25.5	47.6	57.9
DIST										
NoLoc	40.1	53.6	66.1	53.5	55.8	51.9	51.0	47.7	74.3	51.9
Dev	42.7	58.1	71.9	62.9	56.3	55.0	55.1	47.4	73.5	58.9
Test	46.7	58.1	70.2	61.5	58.7	53.7	59.7	57.8	66.0	59.9
NIST										
NoLoc	7.39	13.0	11.4	12.1	9.18	8.64	8.54	7.27	8.92	13.6
Dev	7.50	13.4	12.2	12.7	9.40	8.90	8.92	7.45	9.24	14.1
Test	7.15	13.5	12.0	12.7	9.56	8.00	8.70	7.33	9.13	14.2

Table 5: Automatic metric results for combined system on development and test sets, along with ablation results with no locality features (NoLoc) for the dev set.

5 Results

Many of our results were turned in late because of library compatibility issues: in particular, since Kann and Schütze's code is based on an outdated version of Theano, which is difficult to support, we could not port the morphology inflection system to more powerful computing clusters and were thus limited to training on a single unit with only one GPU. Nevertheless, we were careful to perform no further development after the deadline, and the organizers encouraged us to submit results for all of the languages when we could.

Automatic metric results for the combined morphological inflection and linearization system (with rule-based detokenizer) appear in Table 5, along with no-locality ablation results discussed in the next section. Results for the development and test sets were fairly consistent across all three automatic metrics used in the shared task. Based on the test results shared with the participants, our combined system was among the top performers for all languages, with particularly strong BLEU results for Arabic, Czech, Spanish, Finnish, Portuguese and Russian (French, Italian and Dutch may have suffered from undertrained linearization models). Metric scores varied widely across the languages, though the variation was largely consistent with that observed by other participants, suggesting that some languages are more challenging than others for surface realization (or at least more difficult to achieve high metric scores with).

6 Analysis and Discussion

6.1 Morphological Inflection

Compared to previous work in the context of the SIGMORPHON shared tasks, the SRST '18 input and output vocabulary for the morphological inflection system was much larger. Having a larger search space seems to have affected languages non-uniformly. Although we have different feature sets, Spanish and Russian seem to be unaffected where as Dutch and English scores are drastically lower than expected.

At the actual sub-word level, sequence to sequence models are unable to take account of context originating from outside the input sequence. For example, we observe that the model frequently confuses when to use English 'a' and 'an', since the information necessary to make this prediction does not occur within the character sequence for the word. In future work, an architecture that jointly performs linearization and morphological inflection could address this issue.

Another error type seen at the sub-word level is that although the system learns what affixes look like in a given language, it does not always learn exactly how to apply them, as often seen with Russian. For example, when the system should produce UCHENYJ ('scientist)' and instead produces UCHENOGO, it is confusing the adjectival ending of -OGO for the nominal ending -YJ, both of which however mean masculine, singular, and genitive.

	Language									
	ar	cs	en	es	fi	fr	it	nl	pt	ru
Rate	8.25	12.3	2.43	9.21	7.04	4.11	5.16	8.96	10.6	7.16
Recall	16.2	16.9	16.7	13.8	9.57	6.00	3.45	35.7	18.6	26.9
Precision	25.0	62.0	47.1	44.7	52.9	15.8	25.0	34.9	68.8	23.8

Table 6: Non-projective dependency prevalence, recall and precision in the development set.

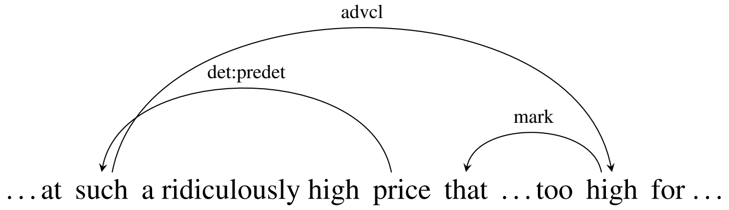
advcl
det:predet
mark
...at such a ridiculously high price that ...too high for...

Figure 4: Example from the development set where a non-projective adverbial clause dependency from *such* to *high* is successfully reproduced, enhancing fluency.

6.2 Linearization

To examine the impact of the dependency locality features, we trained an ablated model with no locality features and compared its performance on the development set to the full model, as shown in Table 5. The ablated model performed worse for all languages with BLEU & NIST, and for most languages with DIST. Moreover, the locality-based features achieved impressive gains in BLEU scores ranging between 2 and 12 points, with the most dramatic gains for Spanish, Russian and English.

We also investigated whether the full model better approximated the total dependency length of the gold development sentences than the ablated model, but the results were inconclusive, with the full model coming closer to gold for some languages but not others. With better incremental features for modeling total dependency length, we plan to investigate in future work whether locality-based features can indeed better match the gold total dependency length in an incremental setting, as found in our earlier work with a bottom-up, chart-based realizer (White and Rajkumar, 2012). Nevertheless, we did find many examples such as the one in Table 7 where the locality-based features helped to ameliorate search errors. In the table, the realization using the ablated model (NoLoc) fails to linearize the dependents *al Sadr - 's* anywhere near their head *Muqtada*, mistakenly leaving them till the end of the sentence where they contribute to a much higher total dependency length than in the gold sentence (Gold) or the realization using the full model (Dev). Note that the full model does not correctly order the name *Muqtada al-Sadr* either, but the realization is still much easier to interpret as intended. As an aside, the realization also includes another local ordering error, *before only three months*; we expect that incorporating a neural language model in future work will resolve many problems of this kind.

Turning now to non-projectivity, we found that sentences with extraposed phrases like those in the gold sentences were sometimes successfully generated. Table 6 shows that the percentage of sentences in the development set with at least one non-projective dependency ranged from a low of 2.5% for English to over 12% for Czech.[3] Recall of the gold non-projective dependencies was generally low, while precision was generally more reasonable, reaching 62% for Czech. Restricting outputs to be projective generally led to small decreases in BLEU scores on the development set, with English and Czech seeing the largest drops of 1.5 and 3.3 points, respectively, though Finnish and Russian witnessed improvements of nearly 1 BLEU point. An example illustrating the successful realization of a non-projective dependency appears in Figure 4; by contrast, if only projective dependencies are allowed, the best possible realization would still be the quite unnatural ... *at a*

[3] A dependency between a head and its dependent was considered projective if all the intervening words (ignoring punctuation tokens) in the linearized sequence were descendants of the head.

Gold: the Coalition decision to provoke a fight with Muqtada al - Sadr 's movement only three months before the Coaliti on Provisional Authority goes out of business has to be seen as a form of gross incompetence in governance . (deplen 84)

NoLoc: the Coalition decision to provoke a fight with Muqtada movement before three months only the Provisional Coalition Authority goes out of business has to be seen as a form of gross incompetence in governance . al Sadr - 's (deplen 144)

Dev: the Coalition decision to provoke a fight with Muqtada - Sadr al 's movement before only three months the Coalition Provisional Authority goes out of business has to be seen as a form of gross incompetence in governance . (deplen 90)

Table 7: Example from the development set showing how locality-based features help ameliorate search errors (with total dependency length in parentheses).

ridiculously high price such that ....

7 Conclusion

We have shown surprisingly competitive results by modeling realization as a two-stage process where we first generate morphologically inflected wordforms using a neural sequence-to-sequence model and then incrementally linearize those wordforms using a global linear model. We additionally show that NMT systems, which have been producing state-of-the-art results in morphological reinflection, can be generalized and integrated into other tasks. We also find that dependency structure and dependency locality are highly informative in the linearization step and allow us to also generate some cases of non-projectivity. In future work, we intend to pursue coupling the learning of morphological inflection and linearization into a single process and using a neural language model to help with linearization choices.

Acknowledgments

We thank Micha Elsner for helpful comments and discussion. This work was supported in part by NSF grants IIS-1319318 and IIS-1618336. The work was also supported by an allocation of computing time from the Ohio Supercomputer Center.

References

Farrell Ackerman, James P Blevins, and Robert Malouf. 2009. Parts and wholes: Implicative patterns in inflectional paradigms. *Analogy in grammar: Form and acquisition*, pages 54–82.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Srinivas Bangalore and Owen Rambow. 2000. Exploiting a probabilistic hierarchical model for generation. In *Proc. COLING-00*.

Otto Behaghel. 1932. *Deutsche Syntax: eine geschichtliche Darstellung. Band IV. Wortstellung. Periodenbau*. Heidelberg: Carl Universitatsbuchhandlung, Germany.

Bernd Bohnet, Simon Mille, Benoît Favre, and Leo Wanner. 2011. Stumaba : From deep representation to surface. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 232–235. Association for Computational Linguistics.

Bernd Bohnet, Leo Wanner, Simon Mill, and Alicia Burga. 2010. Broad coverage multilingual deep sentence generation with a stochastic multi-level realizer. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 98–106. Coling 2010 Organizing Committee.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2015. Gated feedback recurrent neural networks. In *International Conference on Machine Learning*, pages 2067–2075.

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. *CoRR*, abs/1706.09031.

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 2016 Meeting of SIGMORPHON*, Berlin, Germany. Association for Computational Linguistics.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

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

Ondřej Dušek and Filip Jurcicek. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 45–51. Association for Computational Linguistics.

Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2016. Morphological inflection generation using character sequence to sequence learning. In *Proc. of NAACL*.

Katja Filippova and Michael Strube. 2007. Generating constituent order in German clauses. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computer Linguistics.

Katja Filippova and Michael Strube. 2009. Tree linearization in English: Improving language model based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 225–228, Boulder, Colorado. Association for Computational Linguistics.

Edward Gibson. 2000. Dependency locality theory: A distance-based theory of linguistic complexity. In Alec Marantz, Yasushi Miyashita, and Wayne O'Neil, editors, *Image, Language, brain: Papers from the First Mind Articulation Project Symposium*. MIT Press, Cambridge, MA.

Daniel Gildea and David Temperley. 2010. Do grammars minimize dependency length? *Cognitive Science*, 34(2):286–310.

Yuqing Guo, Josef van Genabith, and Haifeng Wang. 2008. Dependency-based n-gram models for general purpose sentence realisation. In *Proc. COLING-08*.

Yuqing Guo, Deirdre Hogan, and Josef van Genabith. 2011. Dcu at generation challenges 2011 surface realisation track. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 227–229. Association for Computational Linguistics.

John A. Hawkins. 1994. *A Performance Theory of Order and Constituency*. Cambridge University Press, New York.

John A. Hawkins. 2004. *Efficiency and Complexity in Grammars*. Oxford University Press.

Katharina Kann and Hinrich Schütze. 2016. MED: The LMU system for the SIGMORPHON 2016 shared task on morphological reinflection. *ACL 2016*, page 62.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157. Association for Computational Linguistics.

Haitao Liu. 2008. Dependency distance as a metric of language comprehension difficulty. *Journal of Cognitive Science*, 9(2):159–191.

Yijia Liu, Yue Zhang, Wanxiang Che, and Bing Qin. 2015. Transition-based syntactic linearization. In *Proceedings of NAACL*, Denver, Colorado, USA.

Simon Mille, Anja Belz, Bernd Bohnet, Yvette Graham, Emily Pitler, and Leo Wanner. 2018. The First Multilingual Surface Realisation Shared Task (SR'18): Overview and Evaluation Results. In *Proceedings of the 1st Workshop on Multilingual Surface Realisation (MSR), 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1–10, Melbourne, Australia.

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. Association for Computational Linguistics.

Ratish Puduppully, Yue Zhang, and Manish Shrivastava. 2016. Transition-based syntactic linearization with lookahead features. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 488–493, San Diego, California. Association for Computational Linguistics.

Rajakrishnan Rajkumar, Marten van Schijndel, Michael White, and William Schuler. 2016. Investigating locality effects and surprisal in written English syntactic choice phenomena. *Cognition*, 155:204–232.

David Temperley. 2007. Minimization of dependency length in written English. *Cognition*, 105(2):300–333.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721. Association for Computational Linguistics.

Michael White and Rajakrishnan Rajkumar. 2012. Minimal dependency length in realization ranking. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 244–255, Jeju Island, Korea. Association for Computational Linguistics.

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

Yue Zhang and Stephen Clark. 2015. Syntax-based word ordering using learning-guided search. *Computational Linguistics*, 41(3).