Experiments with Generative Models for Dependency Tree Linearization

Richard Futrell and Edward Gibson
Department of Brain and Cognitive Sciences
Massachusetts Institute of Technology
{futrell, egibson}@mit.edu

Abstract

We present experiments with generative models for linearization of unordered labeled syntactic dependency trees (Belz et al., 2011; Rajkumar and White, 2014). Our linearization models are derived from generative models for dependency structure (Eisner, 1996). We present a series of generative dependency models designed to capture successively more information about ordering constraints among sister dependents. We give a dynamic programming algorithm for computing the conditional probability of word orders given tree structures under these models. The models are tested on corpora of 11 languages using test-set likelihood, and human ratings for generated forms are collected for English. Our models benefit from representing local order constraints among sisters and from backing off to less sparse distributions, including distributions not conditioned on the head.

1 Introduction

We explore generative models for producing linearizations of unordered labeled syntactic dependency trees. This specific task has attracted attention in recent years (Filippova and Strube, 2009; He et al., 2009; Belz et al., 2011; Bohnet et al., 2012; Zhang, 2013) because it forms a useful part of a natural language generation pipeline, especially in machine translation (Chang and Toutanova, 2007) and summarization (Barzilay and McKeown, 2005). Closely related tasks are generation of sentences given CCG parses (White and Rajkumar, 2012), bags of words (Liu et al., 2015), and semantic graphs (Braune et al., 2014).

Here we focus narrowly on testing probabilistic generative models for dependency tree linearization. In contrast, the approach in most previous work is to apply a variety of scoring functions to trees and linearizations and search for an optimally-scoring tree among some set. The probabilistic linearization models we investigate are derived from generative models for dependency trees (Eisner, 1996), as most commonly used in unsupervised grammar induction (Klein and Manning, 2004; Gelling et al., 2012). Generative dependency models have typically been evaluated in a parsing task (Eisner, 1997). Here, we are interested in the inverse task: inferring a distribution over linear orders given unordered dependency trees.

This is the first work to consider generative dependency models from the perspective of word ordering. The results can potentially shed light on how ordering constraints are best represented in such models. In addition, the use of probabilistic models means that we can easily define well-motivated normalized probability distributions over orders of dependency trees. These distributions are useful for answering scientific questions about crosslinguistic word order in quantitative linguistics, where obtaining robust estimates has proven challenging due to data sparsity (Futrell et al., 2015).

The remainder of the work is organized as follows. In Section 2 we present a set of generative linearization models. In Section 3 we compare the performance of the different models as measured by test-set probability and human acceptability ratings. We also compare our performance with other systems from the literature. Section 4 concludes.

2 Generative Models for Projective Dependency Tree Linearization

We investigate head-outward projective generative dependency models. In these models, an ordered dependency tree is generated by the following kind

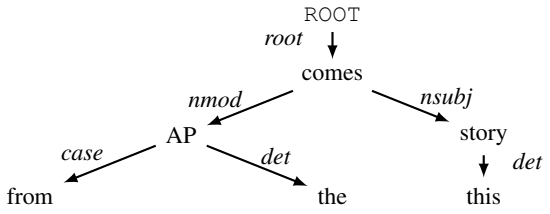
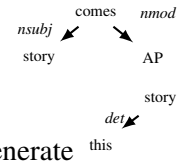


Figure 1: Example unordered dependency tree. Possible linearizations include (1) *This story comes from the AP* and (2) *From the AP comes this story*. Order 2 is the original order in the corpus, but order 1 is much more likely under our models.

of procedure. Given a head node, we use some generative process G to generate a depth-1 subtree rooted in that head node. Then we apply the procedure recursively to each of the dependent nodes. By applying the procedure starting at a ROOT node, we generate a dependency tree. For example, to generate the dependency tree in Figure 1 from the node *comes* down, we take the head *comes* and generate the subtree  , then we take the head *story* and generate , and so on. In this work, we experiment with different specific generative processes G which generate a local subtree conditioned on a head.

2.1 Model Types

Here we describe some possible generative processes G which generate subtrees conditioned on a head. These models contain progressively more information about ordering relations among sister dependents.

A common starting point for G is **Eisner Model C** (Eisner, 1996). In this model, dependents on one side of the head are generated by repeatedly sampling from a categorical distribution until a special stop-symbol is generated. The model only captures the propensity of dependents to appear on the left or right of the head, and does not capture any order constraints between sister dependents on one side of the head.

We consider a generalization of Eisner Model C which we call **Dependent N-gram** models. In a Dependent N-gram model, we generate dependents on each side the head by sampling a *sequence* of dependents from an N-gram model. Each dependent is generated conditional on the $N-1$ previously generated dependents from the head outwards. We have two separate N-gram sequence distributions for left and right dependents. Eisner Model C can be seen as a Dependent N-gram model with $N = 1$.

We also consider a model which can capture many more ordering relations among sister dependents: given a head h, sample a subtree whose head is h from a Categorical distribution over subtrees. We call this the **Observed Orders** model because in practice we are simply sampling one of the observed orders from the training data. This generative process has the capacity to capture the most ordering relations between sister dependents.

2.1.1 Distributions over Permutations of Dependents

We have discussed generative models for ordered dependency trees. Here we discuss how to use them to make generative models for word orders conditional on unordered dependency trees.

Suppose we have a generative process G for dependency trees which takes a head h and generates a sequence of dependents $\mathbf{w}_l$ to the left of h and a sequence of dependents $\mathbf{w}_r$ to the right of h. Let $\mathbf{w}$ denote the pair $(\mathbf{w}_l, \mathbf{w}_r)$, which we call the **configuration** of dependents. To get the probability of some $\mathbf{w}$ given an unordered subtree u, we want to calculate the probability of $\mathbf{w}$ given that G has generated the particular multiset $\mathbf{W}$ of dependents corresponding to u. To do this, we calculate:

$$\begin{aligned} p(\mathbf{w}|\mathbf{W}) &= \frac{p(\mathbf{w}, \mathbf{W})}{p(\mathbf{W})} \\ &= \frac{p(\mathbf{w})}{Z}, \end{aligned} \tag{1}$$

where

$$Z = \sum_{\mathbf{w}' \in \mathcal{W}} p(\mathbf{w}') \tag{2}$$

and $\mathcal{W}$ is the set of all possible configurations $(\mathbf{w}_l, \mathbf{w}_r)$ compatible with multiset $\mathbf{W}$. That is, $\mathcal{W}$ is the set of pairs of permutations of multisets $\mathbf{W}_l$ and $\mathbf{W}_r$ for all possible partitions of $\mathbf{W}$ into $\mathbf{W}_l$ and $\mathbf{W}_r$. The generative dependency model gives us the probability $p(\mathbf{w})$.

It remains to calculate the normalizing constant Z, the sum of probabilities of possible configurations. For the Observed Orders model, Z is the sum of probabilities of subtrees with the same dependents as subtree u. For the Dependent N-gram models of order N, we calculate Z using a dynamic programming algorithm, presented in Al-

gorithm 1 as memoized recursive functions. When $N = 1$ (Eisner Model C), Z is more simply:

$$Z_{\text{emc}} = p_L(\texttt{stop}) \times p_R(\texttt{stop}) \times \sum_{(\mathbf{W}_l, \mathbf{W}_r) \in \text{PARTS}(\mathbf{W})} |\mathbf{W}_l|! \times |\mathbf{W}_r|! \times \prod_{w \in \mathbf{W}_l} p_L(w) \prod_{w \in \mathbf{W}_r} p_R(w), \quad (3)$$

where $\text{PARTS}(\mathbf{W})$ is the set of all partitions of multiset $\mathbf{W}$ into two multisets $\mathbf{W}_l$ and $\mathbf{W}_r$, p_L is the probability mass function for a dependent to the left of the head, p_R is the function for a dependent to the right, and `stop` is a special symbol in the support of p_L and p_R which indicates that generation of dependents should halt. The probability mass functions may be conditional on the head h. These methods for calculating Z make it possible to transform a generative dependency model into a model of dependency tree *ordering* conditional on local subtree structure.

Algorithm 1 Compute the sum of probabilities of all configurations of dependents $\mathbf{W}$ under a Dependent N-gram model with two component N-gram models of order N: p_R for sequences to the right of the head and p_L for sequences to the left.

memoized function RIGHT_NORM($\mathbf{r}$, $\mathbf{c}$)
  if $|\mathbf{r}| = 0$ **then**
    return $p_R(\texttt{stop} \mid \mathbf{c})$
  end if
  $Z \leftarrow 0$
  for $i = 1 : |\mathbf{r}|$ **do**
    $\mathbf{r}' \leftarrow$ elements of $\mathbf{r}$ except the ith
    $\mathbf{c}' \leftarrow$ append r_i to $\mathbf{c}$ then truncate to length $N - 1$
    $Z \leftarrow Z + p_R(r_i|\mathbf{c}) \times \text{RIGHT\_NORM}(\mathbf{r}', \mathbf{c}')$
  end for
  return Z
end memoized function
memoized function LEFT_NORM($\mathbf{r}$, $\mathbf{c}$)
  $Z \leftarrow p_L(\texttt{stop} \mid \mathbf{c}) \times \text{RIGHT\_NORM}([\texttt{start}], \mathbf{r})$
  for $i = 1 : |\mathbf{r}|$ **do**
    $\mathbf{r}' \leftarrow$ elements of $\mathbf{r}$ except the ith
    $\mathbf{c}' \leftarrow$ append r_i to $\mathbf{c}$ then truncate to length $N - 1$
    $Z \leftarrow Z + p_L(r_i|\mathbf{c}) \times \text{LEFT\_NORM}(\mathbf{r}', \mathbf{c}')$
  end for
  return Z
end memoized function
Result is LEFT_NORM($\mathbf{W}$, [`start`])

2.2 Labelling

The previous section discussed the question of the structure of the generative process for dependency trees. Here we discuss an orthogonal modeling question, which we call **labelling**: what information about the *labels* on dependency tree nodes and edges should be included in our models. Dependency tree nodes are labeled with wordforms, lemmas, and parts-of-speech (POS) tags; and dependency tree edges are labeled with relation types. A model might generate orders of dependents conditioned on all of these labels, or a subset of them. For example, a generative dependnecy model might generate (relation type, dependent POS tag) tuples conditioned on the POS tag of the head of the phrase. When we use such a model for dependency linearization, we would say the model's labelling is relation type, dependent POS, and head POS. In this study, we avoid including wordforms or lemmas in the labelling, to avoid data sparsity issues.

2.3 Model Estimation and Smoothing

In order to alleviate data sparsity in fitting our models, we adopt two smoothing methods from the language modelling literature.

All categorical distributions are estimated using add-k smoothing where $k = 0.01$. For the Dependent N-gram models, this means adding k pseudocounts for each possible dependent in each context. For the Observed Orders model, this means adding k pseudocounts for each possible permutation of the head and its dependents.

We also experiment with combining our models into mixture distributions. This can be viewed as a kind of back-off smoothing (Katz, 1987), where the Observed Orders model is the model with the most context, and Dependent N-grams and Eisner Model C are backoff distributions with successively less context. Similarly, models with less information in the labelling can serve as backoff distributions for models with more information in the labelling. For example, a model which is conditioned on the POS of the head can be backed off to a model which does not condition on the head at all. We find optimal mixture weights using the Baum-Welch algorithm tuned on a held-out development set.

3 Evaluation

Here we empirically evaluate some options for model type and model labelling as described above. We are interested in how many of the possible orders of a sentence our model can generate (recall), and in how many of our generated orders really are acceptable (precision). As a recall-like measure, we quantify the probability of the word orders of held-out test sentences. Low probabil-

Labelling	Model	Basque	Czech	English	Finnish	French	German	Hebrew	Indonesian	Persian	Spanish	Swedish
HDR	oo	-6.83	-7.58	-5.23	-7.35	-10.86	-8.36	-9.74	-8.99	-10.39	-11.31	-8.83
	n1	-6.12	-8.97	-5.08	-7.15	-11.54	-9.81	-9.63	-8.68	-10.63	-13.19	-8.37
	n2	-4.86	-6.35	-2.87	-5.30	-6.86	-6.60	-5.91	-5.98	-5.54	-7.47	-4.92
	n3	-5.92	-6.59	-3.13	-5.68	-7.34	-7.02	-6.81	-6.69	-6.49	-8.06	-5.68
	n123	-4.58	-6.18	-2.60	-5.11	-6.67	-6.19	-5.77	-5.73	-5.51	-7.36	-4.72
	oo+n123	-4.52	-5.95	-2.57	-5.04	-6.58	-5.92	-5.68	-5.68	-5.47	-7.27	-4.68
HDR+R	oo	-5.56	-6.78	-3.94	-6.25	-9.63	-7.42	-7.95	-7.51	-9.19	-9.54	-7.28
	n1	-6.08	-8.97	-5.07	-7.16	-11.54	-9.79	-9.58	-8.67	-10.62	-13.17	-8.35
	n2	-4.49	-6.31	-2.62	-5.17	-6.79	-6.34	-5.62	-5.67	-5.42	-7.40	-4.67
	n3	-4.86	-6.41	-2.61	-5.20	-7.08	-6.43	-6.07	-6.02	-6.04	-7.70	-5.02
	n123	-4.41	-6.15	-2.48	-5.01	-6.59	-5.99	-5.54	-5.53	-5.42	-7.29	-4.53
	oo+n123	**-4.29**	**-5.84**	**-2.44**	**-4.88**	**-6.50**	**-5.74**	**-5.40**	**-5.47**	**-5.38**	**-7.09**	**-4.46**

Table 1: Average log likelihood of word order per sentence in test set under various models. Under "Labelling", **HDR** means conditioning on Head POS, Dependent POS, and Relation Type, and **R** means conditioning on Relation Type alone (see Section 2.2). Under "Model", **oo** is the Observed Orders model, **n1** is the Dependent 1-gram model (Eisner Model C), **n2** is the Dependent 2-gram model, and **n3** is the Dependent 3-gram model (see Section 2.1). In both columns, x**+**y means a mixture of model x and model y; **n123** means n1+n2+n3.

ities assigned to held-out sentences indicate that there are possible orders which our model is missing. As a precision-like measure, we get human acceptability ratings for sentence reorderings generated by our model.

We carry out our evaluations using the dependency corpora of the Universal Dependencies project (v1.1) (Agić et al., 2015), with the train/dev/test splits provided in that dataset. We remove nodes and edges dealing with punctuation. Due to space constraints, we only present results from 11 languages here.

3.1 Test-Set Probability

Here we calculate average probabilities of word orders per sentence in the test set. This number can be interpreted as the (negative) average amount of information contained in the word order of a sentence beyond information about dependency relations.

The results for selected languages are shown in Table 1. The biggest gains come from using Dependent N-gram models with $N > 1$, and from backing off the model labelling. The Observed Orders model does poorly on its own, likely due to data sparsity; its performance is much improved when backing off from conditioning on the head. Eisner Model C (n1) also performs poorly, likely because it cannot represent any ordering constraints among sister dependents. The fact it helps to back off to distributions not conditioned on the head suggests that there are commonalities among distributions of dependents of different heads, which could be exploited in further generative dependency models.

3.2 Human Evaluation

We collected human ratings for sentence reorderings sampled from the English models from 54 native American English speakers on Amazon Mechanical Turk. We randomly selected a set of 90 sentences from the test set of the English Universal Dependencies corpus. We generated a reordering of each sentence according to each of 12 model configurations in Table 1. Each participant saw an original sentence and a reordering of it, and was asked to rate how natural each version of the sentence sounded, on a scale of 1 to 5. The order of presentation of the original and reordered forms was randomized, so that participants were not aware of which form was the original and which was a reordering. Each participant rated 56 sentence pairs. Participants were also asked whether the two sentences in a pair meant the same thing, with "can't tell" as a possible answer.

Table 2 shows average human acceptability ratings for reorderings, and the proportion of sentence pairs judged to mean the same thing. The original sentences have an average acceptability rating of 4.48/5. The very best performing models are those which do not back off to a distribution not conditioned on the head. However, in the case of the Observed Orders and other sparse models, we see consistent improvement from this backoff.

Figure 2 shows the acceptability ratings (out of 5) plotted against test set probability. We see that

Labelling	Model	Acceptability	Same Meaning
HDR	oo	2.92	0.58
	n1	2.06	0.44
	n2	3.42	0.78
	n3	3.48	**0.85**
	n123	**3.56**	0.79
	oo+n123	3.45	0.75
HDR+R	oo	3.11	0.72
	n1	2.11	0.49
	n2	3.32	0.80
	n3	3.52	0.77
	n123	3.31	0.76
	oo+n123	3.43	0.80

Table 2: Mean acceptability rating out of 5, and proportion of reordered sentences with the same meaning as the original, for English models. Labels as in Table 1.

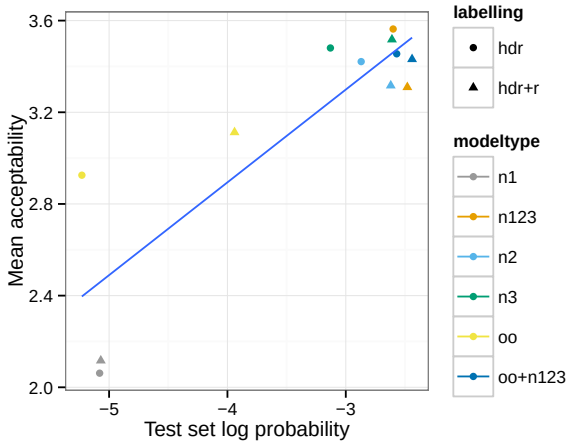


Figure 2: Comparison of test set probability (Table 1) and acceptability ratings (Table 2) for English across models. A least-squares linear regression line is shown. Labels as in Table 1.

the models which yield poor test set probability also have poor acceptability ratings.

3.3 Comparison with other systems

Previous work has focused on the ability to correctly reconstruct the word order of an observed dependency tree. Our goal is to explicitly model a distribution over possible orders, rather than to recover a single correct order, because many orders are often possible, and the particulator order that a dependency tree originally appeared in might not be the most natural. For example, our models typically reorder the sentence "From the AP comes this story" (in Figure 1) as "This story comes from the AP"; the second order is arguably more natural, though the first is idiomatic for this particular phrase. So we do not believe that BLEU scores and other metrics of similarity to a "correct" ordering are particularly relevant for our task.

Previous work uses BLEU scores (Papineni et al., 2002) and human ratings to evaluate generation of word orders. To provide some comparability with previous work, we report BLEU scores on the 2011 Shared Task data here. The systems reported in Belz et al. (2011) achieve BLEU scores ranging from 23 to 89 for English; subsequent work achieves BLEU scores of 91.6 on the same data (Bohnet et al., 2012). Drawing the highest-probability orderings from our models, we achieve a top BLEU score of 57.7 using the model configuration hdr/oo. Curiously, hdr/oo is typically the worst model configuration in the test set probability evaluation (Section 3.1). The BLEU performance is in the middle range of the Shared Task systems. The human evaluation of our models is more optimistic: the best score for Meaning Similarity in the Shared Task was 84/100 (Bohnet et al., 2011), while sentences ordered according to our models were judged to have the same meaning as the original in 85% of cases (Table 2), though these figures are based on different data. These comparisons suggest that these generative models do not provide state-of-the-art performance, but do capture some of the same information as previous models.

3.4 Discussion

Overall, the most effective models are the Dependent N-gram models. The naive approach to modeling order relations among sister dependents, as embodied in the Observed Orders model, does not generalize well. The result suggests that models like the Dependent N-gram model might be effective as generative dependency models.

4 Conclusion

We have discussed generative models for dependency tree linearization, exploring a path less traveled by in the dependency linearization literature. We believe this approach has value for answering scientific questions in quantitative linguistics and for better understanding the linguistic adequacy of generative dependency models.

Acknowledgments

We thank William P. Li, Kyle Mahowald, and Tim O'Donnell for helpful discussions, and Michael White for help accessing data.

References

Željko Agić, Maria Jesus Aranzabe, Aitziber Atutxa, Cristina Bosco, Jinho Choi, Marie-Catherine de Marneffe, Timothy Dozat, Richárd Farkas, Jennifer Foster, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Jan Hajič, Anders Trærup Johannsen, Jenna Kanerva, Juha Kuokkala, Veronika Laippala, Alessandro Lenci, Krister Lindén, Nikola Ljubešić, Teresa Lynn, Christopher Manning, Héctor Alonso Martínez, Ryan McDonald, Anna Missilä, Simonetta Montemagni, Joakim Nivre, Hanna Nurmi, Petya Osenova, Slav Petrov, Jussi Piitulainen, Barbara Plank, Prokopis Prokopidis, Sampo Pyysalo, Wolfgang Seeker, Mojgan Seraji, Natalia Silveira, Maria Simi, Kiril Simov, Aaron Smith, Reut Tsarfaty, Veronika Vincze, and Daniel Zeman. 2015. Universal dependencies 1.1. LINDAT/CLARIN digital library at Institute of Formal and Applied Linguistics, Charles University in Prague.

Regina Barzilay and Kathleen R McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Anja Belz, Michael White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. 2011. The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 217–226.

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.

Bernd Bohnet, Anders Björkelund, Jonas Kuhn, Wolfgang Seeker, and Sina Zarrieß. 2012. Generating non-projective word order in statistical linearization. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 928–939.

Fabienne Braune, Daniel Bauer, and Kevin Knight. 2014. Mapping between english strings and reentrant semantic graphs. In *Int. Conf. on Language Resources and Evaluation (LREC)*.

Pi-Chuan Chang and Kristina Toutanova. 2007. A discriminative syntactic word order model for machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, page 9.

Jason M Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th Conference on Computational Linguistics*, pages 340–345.

Jason M Eisner. 1997. An empirical comparison of probability models for dependency grammar. Technical report, IRCS Report 96–11, University of Pennsylvania.

Katja Filippova and Michael Strube. 2009. Tree linearization in English: Improving language model based approaches. In *Proceedings of NAACL-HLT (Short Papers)*.

Richard Futrell, Kyle Mahowald, and Edward Gibson. 2015. Quantifying word order freedom in dependency corpora. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 91–100, Uppsala, Sweden.

Douwe Gelling, Trevor Cohn, Phil Blunsom, and Joao Graça. 2012. The pascal challenge on grammar induction. In *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 64–80.

Wei He, Haifeng Wang, Yuqing Guo, and Ting Liu. 2009. Dependency based Chinese sentence realization. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 809–816.

Slava M Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(3):400–401.

Dan Klein and Christopher D Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, page 478.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Rajakrishnan Rajkumar and Michael White. 2014. Better surface realization through psycholinguistics. *Language and Linguistics Compass*, 8(10):428–448.

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.

Yue Zhang. 2013. Partial-tree linearization: generalized word ordering for text synthesis. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 2232–2238. AAAI Press.