An experimental analysis of Noise-Contrastive Estimation: the noise distribution matters

Matthieu Labeau and **Alexandre Allauzen**
LIMSI, CNRS, Univ. Paris-Sud, Université Paris-Saclay
Rue John von Neumann, 91403 Orsay cedex, France
{matthieu.labeau, alexandre.allauzen}@limsi.fr

Abstract

Noise Contrastive Estimation (NCE) is a learning procedure that is regularly used to train neural language models, since it avoids the computational bottleneck caused by the output softmax. In this paper, we attempt to explain some of the weaknesses of this objective function, and to draw directions for further developments. Experiments on a small task show the issues raised by the unigram noise distribution, and that a context dependent noise distribution, such as the bigram distribution, can solve these issues and provide stable and data-efficient learning.

1 Introduction

Statistical language models (LMs) play an important role in many tasks, such as machine translation and speech recognition. Neural models, with various neural architectures (Bengio et al., 2001; Mikolov et al., 2010; Chelba et al., 2014; Józefowicz et al., 2016), have recently achieved great success. However, most of these neural architectures have a common issue: large output vocabularies cause a computational bottleneck due to the output normalization.

Different solutions have been proposed, as *shortlists* (Schwenk, 2007), *hierarchical softmax* (Morin and Bengio, 2005; Mnih and Hinton, 2009; Le et al., 2011), or self-normalisation techniques (Devlin et al., 2014; Andreas et al., 2015; Chen et al., 2016). Sampling-based techniques explore a different solution, where a limited number of negative examples are sampled to reduce the normalization cost. The resulting model is theoretically unnormalized. Apart from importance sampling (Bengio and Sénécal, 2008; Jean et al., 2015), the noise contrastive estimation (NCE) provides a simple and efficient sampling strategy, which our work focuses on.

Introduced by (Gutmann and Hyvärinen, 2010), NCE proposes an objective function that replaces the conventional log-likelihood by a binary classification task, discriminating between the real examples provided by the data, and negative examples sampled from a chosen noise distribution. This allows the model to learn indirectly from the data distribution. NCE was first applied to language modeling by (Mnih and Teh, 2012), and then to various models, often in the context of machine translation (Vaswani et al., 2013; Baltescu and Blunsom, 2015; Zoph et al., 2016). However, recently, a comparative study of methods for training large vocabulary LMs (Chen et al., 2016) highlighted the inconsistency of NCE training when dealing with very large vocabularies, showing very different perplexity results for close loss values. In another work (Józefowicz et al., 2016), NCE was shown far less data-efficient than the theoretically similar importance sampling.

In this paper, we focus on a small task to provide an in-depth analysis of the results. NCE relies on the definition of an artificial classification task that must be monitored. Indeed, using a unigram noise distribution as usually advised leads to an ineffective solution, where the model almost systematically classifies words in the noise class. This can be explained by the inability to sample rare words from the noise distribution, yielding inconsistent updates for the most frequent words. We explore other noise distributions and show that designing a more suitable classification task, with for instance a simple bigram distribution, can efficiently correct the weaknesses of NCE.

Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pages 15–20,
Valencia, Spain, April 3-7, 2017.

2 Theoretical background

A neural probabilistic language model with parameters θ outputs, for an input context H, a conditional distribution P_θ^H for the next word, over the vocabulary $\mathcal{V}$. This conditional distribution is defined using the *softmax* activation function:

$$P_\theta^H(w) = \frac{\exp s_\theta(w, H)}{\sum_{w' \in \mathcal{V}} \exp s_\theta(w', H)} \tag{1}$$

Here, $s_\theta(w, H)$ is a scoring function which depends on the network architecture. The denominator is the partition function $Z(H)$, which is used to ensure output scores are normalized into a probability distribution.

2.1 Maximum likelihood training

Maximum likelihood training is realized by minimizing the negative log-likelihood. Parameter updates will be made using this objective gradient

$$\begin{aligned}\frac{\partial}{\partial\theta} \log P_\theta^H(w) = &\frac{\partial}{\partial\theta} s_\theta(w, H) \\ &- \sum_{w' \in \mathcal{V}, w' \neq w} P_\theta^H(w') \frac{\partial}{\partial\theta} s_\theta(w', H)\end{aligned} \tag{2}$$

increasing the positive output's score, while decreasing the score of the rest of the vocabulary. Unfortunately, both output normalization and gradient computation require computation of the score for every word in $\mathcal{V}$, which is the bottleneck during training, since it implies product of very large matrices ($|\mathcal{V}|$ being usually anywhere from tens to hundreds of thousand words).

2.2 Noise contrastive estimation

The idea behind noise contrastive estimation is to learn the relative description of the data distribution P_d to a reference noise distribution P_n, by learning their ratio P_d/P_n. This is done by drawing samples from the noise distribution and learning to discriminate between the two sets via a classification task. Considering a mixture of the data and noise distribution, for each example w with a context H from the data $\mathcal{D}$, we draw k noise samples from P_n^H. With the logistic regression, we want to estimate the posterior probability of which class C ($C = 1$ for the data, $C = 0$ for the noise) the sample comes from. Since we want to approach the data distribution with our model of parameters θ the conditional class probabilities are:

$$P^H(w|C = 1) = P_\theta^H(w)$$

and

$$P^H(w|C = 0) = P_n^H(w)$$

which gives posterior class probabilities:

$$P^H(C = 1|w) = \frac{P_\theta^H(w)}{P_\theta^H(w) + kP_n^H(w)} \tag{3}$$

which can be rewritten as:

$$P^H(C = 1|w) = \sigma_k\left(\log \frac{P_\theta^H(w)}{P_n^H(w)}\right) \tag{4}$$

with:

$$\sigma_k(u) = \frac{1}{1 + k\exp(-u)}$$

The reformulation obtained in equation 4 shows that training a classifier based on a logistic regression will estimate the log-ratio of the two distributions. This allows the learned distribution to be unnormalized, as the partition function is parametrized separately. A normalizing parameter c^H is added, as following:

$$P_\theta^H(w) = s_{\theta_0}(w, H)\exp(c^H)$$

However, this parametrization is context-dependent. In (Mnih and Teh, 2012), the authors argue that these context-dependent parameters c^H can be put to zero, and that given the number of free parameters, the output scores for each context $s_{\theta_0}(\bullet, H)$ will self-normalize.

The objective function is given by maximizing the log-likelihood of the true example w to belong to class $C = 1$ and the noise samples $(w_j^n)_{1 \leq j \leq k}$ to $C = 0$, which is, for one true example[1]:

$$\begin{aligned}J_\theta^H(w) = &\log \frac{s_\theta(w, H)}{s_\theta(w, H) + kP_n^H(w)} \\ &+ \sum_{1 \leq j \leq k} \log \frac{kP_n^H(w_j^n)}{s_\theta(w_j^n, H) + kP_n^H(w_j^n)}\end{aligned} \tag{5}$$

In order to obtain the global objective to maximize, we sum on all examples $(H, w) \in \mathcal{D}$:

$$J_\theta = \sum_{H, w \in \mathcal{D}} J_\theta^H(w) \tag{6}$$

[1] We keep the notation $s_\theta(w, H)$ instead of $s_{\theta_0}(w, H)$ for readability.

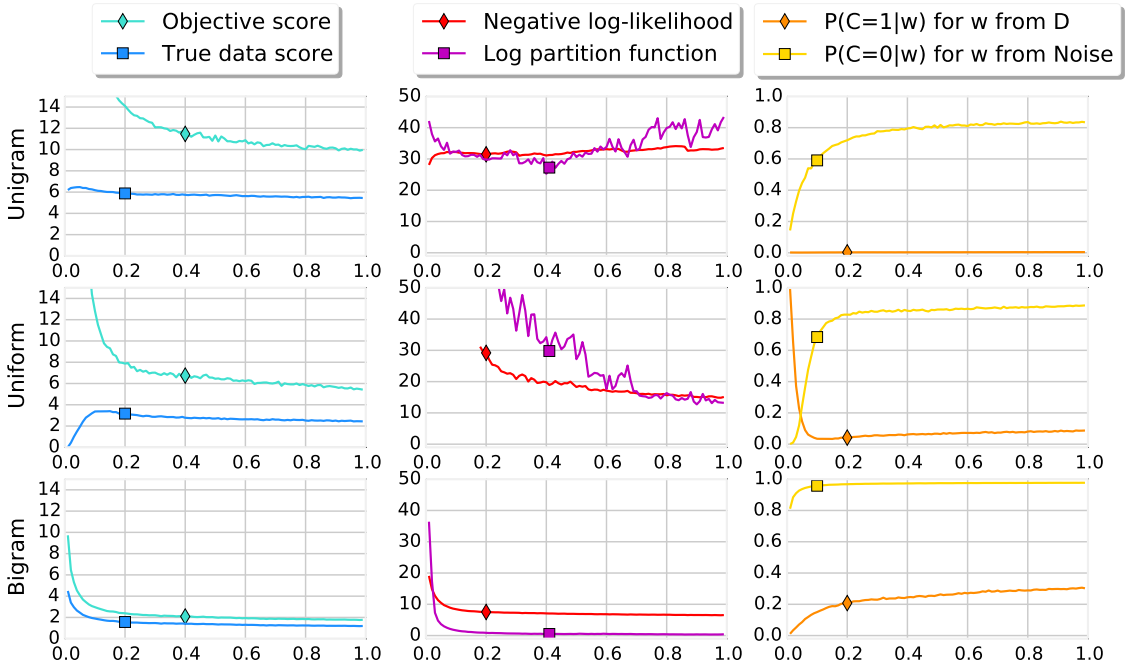

Figure 1: Comparative training of 3-grams neural language models with $k = 25$ noise samples by positive example, with the unigram, uniform, and bigram distribution as noise distributions. Data are recorded over the first epoch. In the first column are shown minus the NCE score, and its fraction concerning true data. In the middle, are shown the negative log-likelihood and the log of the partition function. In the last column, are shown the mean posterior probabilities of classifying data as data, and noise as noise.

3 Experimental set-up

Noise contrastive estimation offers theoretical guarantees (Gutmann and Hyvärinen, 2010). First, the maximum for a global objective defined on an unlimited amount of data is reached for $s_{\theta^*} = \log P_d$, and is the only extrema under mild conditions on the noise distribution. Secondly, the parameters that maximize our experimental objective converge to θ^* in probability as the amount of data grows. Finally, as the number k of noise samples by example increases, the choice of the noise distribution P_n has less impact on the estimation accuracy. Still, the noise distribution should be chosen close to the data distribution, to avoid a too simplistic classification task which could stop the learning process too early. To a certain extent, we can describe it as a trade-off between the number of samples and the effort we need to put on a 'good' noise distribution.

Considering these properties, we investigate the impact of the noise distribution on the training of language models. (Mnih and Teh, 2012) experimented with uniform and unigram distributions, while most of the subsequent literature used the unigram, excepted for (Zoph et al., 2016), who used the uniform with a very large vocabulary.

To monitor the training process with Noise-contrastive estimation, we report the average negative log-likelihood of the model, and its average log-partition function ($\frac{1}{|\mathcal{D}|} \sum_{(H,w)\in\mathcal{D}} \log Z(H)$). In addition to the NCE score, we consider its *true data* term, defined by $\log \frac{s_\theta(w,H)}{s_\theta(w,H)+kP_n^H(w)}$, which quantifies how well the model is able to recognize examples from true data as such, and can be used to estimate the posterior probabilities of each class during training (as described in equation 3).

Training was made on a relatively short English corpus (*news-commentary 2012*) of 4.2M words with a full vocabulary of $\sim$ 70K words. We trained simple feed-forward n-grams neural language models with Tensorflow (Abadi et al., 2015)[2]. Results are recorded on the training data[3].

[2] As our goal is not performance, we choose a simple and time-efficient model, with a context of 3 words, one hidden layer, and embedding and hidden dimension of 50 and 100.

[3] We use a validation set to avoid overfitting.

4 Experiments and Results

The first series of experiments compares different choices of noise distribution (uniform, unigram and bigram) for various vocabulary sizes (from ∼25K to the full vocabulary of ∼70K words). Figure 1 gathers the evolution of different quantities observed during the first training epoch when selecting all words appearing more than once (∼40K words). The same trend is observed for all vocabulary sizes.

For the three noise distributions, the NCE score seems to converge. However, for the unigram distribution, the log-partition function does not decrease, thus neither does the log-likelihood. Interestingly, the posterior classification probabilities shown in the third column reveal a very ineffective behaviour: almost all the positive examples are classified in the noise class.

On the contrary, the use of the uniform distribution yields more consistent results, despite the fact that it is slow to learn.

Finally, learning with the bigram noise distribution shows a very consistent behaviour with a log partition function converging steadily to zero, as well as a negative log-likelihood on par with MLE training. It is moreover very data-efficient, compared to the uniform distribution.

k	25	100	200	500
Uniform	20.9	10.5	8.1	7.1
Unigram	29.7	32.9	30.5	18.5
Unigram ($\alpha = 0.25$)	25.0	8.1	6.9	6.6
Bigram	6.6	6.5	6.5	6.5

Table 1: Negative log-likelihood after one epoch of training with a full vocabulary, for various noise distributions and a varying number of noise samples k

Table 1 shows the negative log-likelihood reached after one epoch of training, for a varying number of noise samples. For the sake of efficiency with context-independant noise distributions, we used for these experiments the NCE implementation native to Tensorflow, for which the noise samples are re-used for all the positive examples in the training batch. While this certainly lowers the performance of the algorithm, we believe it still demonstrates how importantly the convergence speed is impacted by the number of noise samples for context-independant noise distributions, compared to the bigram distribution.

However, using the bigram distribution implies to maintain bigram counts. This can be costly with a large vocabulary size, but not prohibitive. We thus make further experiments with context-independent noise distributions.

A common trick, when using any kind of negative sampling, is to employ a distortion coefficient $0 < \alpha < 1$ to smooth the unigram distribution, by raising every count $c(w)$ to $c(w)^\alpha$, as it is done in (Mikolov et al., 2013). We can then try to get the 'good' of each distribution, which is a balance between the sampling of frequent and rare words as noise, while staying close to the data. Results are shown on figure 2. Distortion heavily influences how the model converges: being closer to the uniform distribution makes training easier, while retaining the unigram distribution's shape is still needed. This is also shown in table 1.

To get a better idea of the differences between those distributions, we first examine the ability of the models to recognize positive examples as such for a portion of the vocabulary containing the most frequent words. The two top graphs of figure 3 show that both the uniform and a distorted unigram distribution help the model to learn to classify the 1000 most frequent words, while almost no information seems to be kept on the rest (which represents $\sim \frac{1}{4}$ of the training data). However, the model using a distorted unigram seems a little more balanced in what it learns, for about the same average performance. The third graph shows that its log-partition function is behaving quite better, which explains the negative log-likelihood gap observed in figure 2 between these two distributions.

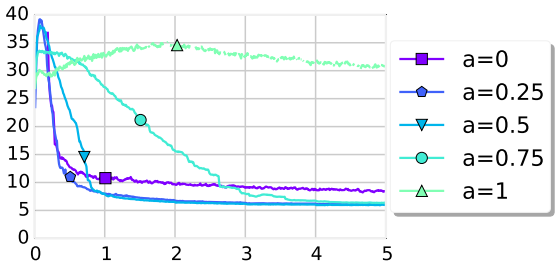

Figure 2: Comparative training of full vocabulary models with $k = 100$ noise samples for a varying distortion, on 5 epochs.

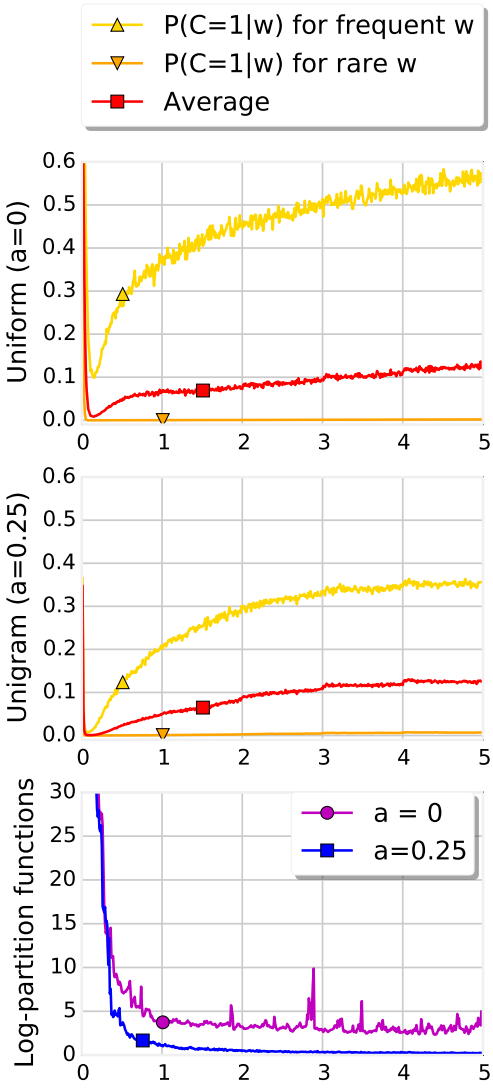

Figure 3: Mean posterior probabilities of recognizing true examples coming from the training data as such, for the 1K most frequent words, the rest of the vocabulary, and the average, for a uniform and a unigram distribution with distortion. The bottom graph shows the two log-partition functions. Training is done on full vocabulary models, with $k = 100$ noise samples, on 5 epochs.

These results show how changing the shape of the noise distribution can positively affect training: using distortion allows to smooth the unigram distribution, avoiding to sample only frequent words, while reaching a better negative log-likelihood than with a uniform distribution. However, as indicated by table 1, models trained with a bigram noise distribution need far less noise samples or data.

5 Conclusion

Given the difficulty to train neural language models with NCE for large vocabularies, this paper aimed to get a better understanding of its mechanisms and weaknesses. Our results indicate that the theoretical trade-off between the number of noise samples and the effort we need to put on a 'good' noise distribution is verified in practice. It also impacts the quantity of training data required, and the training stability. Notably, a context dependent noise distribution yields a satisfactory classification task, along with a faster and steadier training. In the future, we project to work on an intermediate context-dependent noise distribution, which would be able to scale well with large vocabularies.

Acknowledgments

We wish to thank the anonymous reviewers for their helpful comments. This work has been partly funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 645452 (QT21).

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Jacob Andreas, Maxim Rabinovich, Michael I. Jordan, and Dan Klein. 2015. On the accuracy of self-normalized log-linear models. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1783–1791.

Paul Baltescu and Phil Blunsom. 2015. Pragmatic neural language modelling in machine translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 820–829, Denver, Colorado, May–June. Association for Computational Linguistics.

Yoshua Bengio and Jean-Sébastien Sénécal. 2008. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Trans. Neural Networks*, 19(4):713–722.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2001. A neural probabilistic language model.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2014. One billion word benchmark for measuring progress in statistical language modeling. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 2635–2639.

Wenlin Chen, David Grangier, and Michael Auli. 2016. Strategies for training large vocabulary neural language models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1975–1985, Berlin, Germany, August. Association for Computational Linguistics.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1370–1380, Baltimore, Maryland, June. Association for Computational Linguistics.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, pages 297–304.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Beijing, China, July. Association for Computational Linguistics.

Rafal Józefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *CoRR*, abs/1602.02410.

Hai-Son Le, Ilya Oparin, Alexandre Allauzen, Jean-Luc Gauvain, and Francois Yvon. 2011. Structured output layer neural network language model. In *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, pages 5524–5527, Prague, Czech Republic.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Andriy Mnih and Geoffrey E Hinton. 2009. A scalable hierarchical distributed language model. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1081–1088. Curran Associates, Inc.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 246–252. Society for Artificial Intelligence and Statistics.

Holger Schwenk. 2007. Continuous space language models. *Comput. Speech Lang.*, 21(3):492–518, July.

Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. 2013. Decoding with large-scale neural language models improves translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1387–1392, Seattle, Washington, USA, October. Association for Computational Linguistics.

Barret Zoph, Ashish Vaswani, Jonathan May, and Kevin Knight. 2016. Simple, fast noise-contrastive estimation for large rnn vocabularies. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1217–1222, San Diego, California, June. Association for Computational Linguistics.