An Analysis of Simple Data Augmentation for Named Entity Recognition

Xiang Dai[1,2,3] **Heike Adel**[1]
[1]Bosch Center for Artificial Intelligence, Renningen, Germany
[2]University of Sydney, Sydney, Australia
[3]CSIRO Data61, Sydney, Australia
dai.dai@csiro.au heike.adel@de.bosch.com

Abstract

Simple yet effective data augmentation techniques have been proposed for sentence-level and sentence-pair natural language processing tasks. Inspired by these efforts, we design and compare data augmentation for named entity recognition, which is usually modeled as a token-level sequence labeling problem. Through experiments on two data sets from the biomedical and materials science domains (i2b2-2010 and MaSciP), we show that simple augmentation can boost performance for both recurrent and transformer-based models, especially for small training sets.

1 Introduction

Modern deep learning techniques typically require a lot of labeled data (Bowman et al., 2015; Conneau et al., 2017). However, in real-world applications, such large labeled data sets are not always available. This is especially true in some specific domains, such as the biomedical and materials science domain, where annotating data requires expert knowledge and is usually time-consuming (Karimi et al., 2015; Friedrich et al., 2020). Different approaches have been investigated to solve this low-resource problem. For example, transfer learning pretrains language representations on self-supervised or rich-resource source tasks and then adapts these representations to the target task (Ruder, 2019; Gururangan et al., 2020). Data augmentation expands the training set by applying transformations to training instances without changing their labels (Wang and Perez, 2017).

Recently, there is an increased interest on applying data augmentation techniques on sentence-level and sentence-pair natural language processing (NLP) tasks, such as text classification (Wei and Zou, 2019; Xie et al., 2019), natural language inference (Min et al., 2020) and machine translation (Wang et al., 2018). Augmentation methods explored for these tasks either create augmented instances by manipulating a few words in the original instance, such as word replacement (Zhang et al., 2015; Wang and Yang, 2015; Cai et al., 2020), random deletion (Wei and Zou, 2019), or word position swap (Şahin and Steedman, 2018; Min et al., 2020); or create entirely artificial instances via generative models, such as variational auto encoders (Yoo et al., 2019; Mesbah et al., 2019) or back-translation models (Yu et al., 2018; Iyyer et al., 2018).

Different from these sentence-level NLP tasks, named entity recognition (NER) does predictions on the token level. That is, for each token in the sentence, NER models predict a label indicating whether the token belongs to a mention and which entity type the mention has. Therefore, applying transformations to tokens may also change their labels. Due to this difficulty, data augmentation for NER is comparatively less studied. In this work, we fill this research gap by exploring data augmentation techniques for NER, a token-level sequence labeling problem.

Our contributions can be summarized as follows:

1. We survey previously used data augmentation techniques for sentence-level and sentence-pair NLP tasks and adapt some of them for the NER task.

Proceedings of the 28th International Conference on Computational Linguistics, pages 3861–3867
Barcelona, Spain (Online), December 8-13, 2020

2. We conduct empirical comparisons of different data augmentations using two English domain-specific data sets: MaSciP (Mysore et al., 2019) and i2b2-2010 (Uzuner et al., 2011). Results show that simple augmentation can even improve over a strong baseline with large-scale pretrained transformers.

2 Related Work

In this section, we survey previously used data augmentation methods for NLP tasks, grouping them into four categories:

Word replacement: Various word replacement variants have been explored for text classification tasks. Zhang et al. (2015) and Wei and Zou (2019) replace words with one of their synonyms, retrieved from an English thesaurus (e.g., WordNet). Kobayashi (2018) replace words with other words that are predicted by a language model at the word positions. Xie et al. (2019) replace uninformative words with low TF-IDF scores with other uninformative words for topic classification tasks.

For machine translation, word replacement has also been used to generate additional parallel sentence pairs. Wang et al. (2018) replace words in both the source and the target sentence by other words uniformly sampled from the source and the target vocabularies. Fadaee et al. (2017) search for contexts where a common word can be replaced by a low-frequency word, relying on recurrent language models. Gao et al. (2019) replace a randomly chosen word by a soft word, which is a probabilistic distribution over the vocabulary, provided by a language model.

In addition, there are two special word replacement cases, inspired by dropout and masked language modeling: replacing a word by a zero word (i.e., dropping entire word embeddings) (Iyyer et al., 2015), or by a [MASK] token (Wu et al., 2018).

Mention replacement: Raiman and Miller (2017) augment a question answering training set using an external knowledge base. In particular, they extract nominal groups in the training set, perform string matching with entities in Wikidata, and then randomly replace them with other entities of the same type. In order to remove gender bias from coreference resolution systems, Zhao et al. (2018) propose to generate an auxiliary dataset where all male entities are replaced by female entities, and vice versa, using a rule-based approach.

Swap words: Wei and Zou (2019) randomly choose two words in the sentence and swap their positions to augment text classification training sets. Min et al. (2020) explore syntactic transformations (e.g., subject/object inversion) to augment the training data for natural language inference. Şahin and Steedman (2018) rotate tree fragments around the root of the dependency tree to form a synthetic sentence and augment low-resource language part-of-speech tagging training sets.

Generative models: Yu et al. (2018) train a question answering model with data generated by back-translation from a neural machine translation model. Kurata et al. (2016) and Hou et al. (2018) use a sequence-to-sequence model to generate diversely augmented utterances to improve the dialogue language understanding module. Xia et al. (2019) convert data from a high-resource language to a low-resource language, using a bilingual dictionary and an unsupervised machine translation model in order to expand the machine translation training set for the low-resource language.

3 Data Augmentation for NER

Inspired by the efforts described in Section 2, we design several simple data augmentation methods for NER. Note that these augmentations do not rely on any externally trained models, such as machine translation models or syntactic parsing models, which are by themselves difficult to train in low-resource domain-specific scenarios.

Label-wise token replacement (LwTR): For each token, we use a binomial distribution to randomly decide whether it should be replaced. If yes, we then use a label-wise token distribution, built from the original training set, to randomly select another token with the same label. Thus, we keep the original

	Instance											
None	She	did	not	complain	of	headache	or	any	other	neurological	symptoms	.
	O	O	O	O	O	B-problem	O	B-problem	I-problem	I-problem	I-problem	O
LwTR	L.	One	not	complain	of	headache	he	any	interatrial	neurological	current	.
	O	O	O	O	O	B-problem	O	B-problem	I-problem	I-problem	I-problem	O
SR	She	did	non	complain	of	headache	or	whatsoever	former	neurologic	symptom	.
	O	O	O	O	O	B-problem	O	B-problem	I-problem	I-problem	I-problem	O
MR	She	did	not	complain	of	neuropathic	pain	syndrome	or	acute	pulmonary	disease	.
	O	O	O	O	O	B-problem	I-problem	I-problem	O	B-problem	I-problem	I-problem	O
SiS	not	complain	She	did	of	headache	or	neurological	any	symptoms	other	.	
	O	O	O	O	O	B-problem	O	B-problem	I-problem	I-problem	I-problem	O	

Table 1: Original training instance and different types of augmented instances. We highlight changes using blue color. Note that LwTR (Label-wise token replacement) and SiS (Shuffle within segments) change token sequence only, whereas SR (Synonym replacement) and MR (Mention replacement) may also change the label sequence.

label sequence unchanged. Taking the instance in Table 1 as an example, there are five tokens replaced by other tokens which share the same label with the original tokens.

Synonym replacement (SR): Our second approach is similar to LwTR, except that we replace the token with one of its synonyms retrieved from WordNet. Note that the retrieved synonym may consist of more than one token. However, its BIO-labels can be derived using a simple rule: If the replaced token is the first token within a mention (i.e., the corresponding label is 'B-EntityType'), we assign the same label to the first token of the retrieved multi-word synonym, and 'I-EntityType' to the other tokens. If the replaced token is inside a mention (i.e., the corresponding label is 'I-EntityType'), we assign its label to all tokens of the multi-word synonym.

Mention replacement (MR): For each mention in the instance, we use a binomial distribution to randomly decide whether it should be replaced. If yes, we randomly select another mention from the original training set which has the same entity type as the replacement. The corresponding BIO-label sequence can be changed accordingly. For example, in Table 1, the mention 'headache [B-problem]' is replaced by another problem mention 'neuropathic pain syndrome [B-problem I-problem I-problem]'.

Shuffle within segments (SiS): We first split the token sequence into segments of the same label. Thus, each segment corresponds to either a mention or a sequence of out-of-mention tokens. For example, the original sentence in Table 1 is split into five segments: [She did not complain of], [headache], [or], [any other neurological symptoms], [.]. Then for each segment, we use a binomial distribution to randomly decide whether it should be shuffled. If yes, the order of the tokens within the segment is shuffled, while the label order is kept unchanged.

All: We also explore to augment the training set using all aforementioned augmentation methods. That is, for each training instance, we create multiple augmented instances, one per augmentation method.

4 Experiments and Results

4.1 Datasets

We present an empirical analysis of the data augmentation methods described in Section 3 on two English datasets from the materials science and biomedical domains: MaSciP (Mysore et al., 2019)[1] and i2b2-2010 (Uzuner et al., 2011).[2]

MaSciP contains synthesis procedures annotated with synthesis operations and their typed arguments (e.g., Material, Synthesis-Apparatus, etc.). We use the train-dev-test split provided by the authors. i2b2-

[1]https://github.com/olivettigroup/annotated-materials-syntheses
[2]https://portal.dbmi.hms.harvard.edu/

2010 focuses on the identification of Problem, Treatment and Test from patient reports. We use the train-test split from its corresponding shared task setting and randomly select 15% of sentences from the training set as the development set.

To simulate a low-resource setting, we select the first 50, 150, 500 sentences which contain at least one mention from the training set to create the corresponding small, medium, and large training sets (denoted as S, M, L in Table 3, whereas the complete training set is denoted as F) for each data set. Note that we apply data augmentation only on the training set, without changing the development and test sets.

4.2 Backbone models

We model the NER task as a sequence-labeling task. Let $\mathbf{x} = \langle x_1, \ldots, x_T \rangle$ be a sequence of T tokens, the model aims to predict a label sequence $\mathbf{y} = \langle y_1, \ldots, y_T \rangle$, where each label is composed of a position indicator (e.g., BIO schema) and an entity type. The state-of-the-art sequence-labeling models roughly consist of two components: a neural-based encoder which creates contextualized embeddings r_i for each token, and a conditional random field output layer, which captures dependencies between neighboring labels:

$$\hat{P}(y_{1:T}|r_{1:T}) \propto \prod_{i=1}^{T} \psi_i(y_{i-1}, y_i, r_i).$$

We consider two encoder variants in our study: one based on LSTM (Graves et al., 2013) and one based on BERT (Devlin et al., 2019). The LSTM-based encoder consists of a context-independent token embedding layer (e.g., GloVe (Pennington et al., 2014)) and a bidirectional LSTM layer, whose weights are learned from scratch. The representations r_i are obtained by concatenating the hidden states of the forward and backward LSTMs at each token position. The BERT-based encoder consists of a sub-token embedding layer and a stack of multi-head self-attention and fully-connected feed-forward layers. The final hidden state corresponding to the first sub-token within each token is used as the representation r_i. Studies on domain-specific BERT models show that effectiveness on downstream tasks can be improved when the BERT models are further pretrained on in-domain data (Gururangan et al., 2020; Dai et al., 2020). We thus choose SciBERT (Beltagy et al., 2019), which is pretrained on scholar articles, and fine-tune it on the NER task. In our preliminary experiments, we observe that SciBERT achieves significant better results than BERT (Devlin et al., 2019).

We use the Micro-average string match F_1 score to evaluate the effectiveness of the models. The model which is most effective on the development set, measured using the F_1 score, is finally evaluated on the test set.

4.3 Hyperparameters

For each augmentation method, we tune the number of generated instances per training instance from a list of numbers: {1, 3, 6, 10}. When all data augmentation methods are applied, we reduce this tuning list to: {1, 2, 3}, so that the total number of generated instances given each original training instance is roughly the same for different experiments. We also tune the p value of the binomial distribution which is used to decide whether a token or a mention should be replaced (cf., Section 3). It is searched over the range from 0.1 to 0.7, with an incrementation step of 0.2. We perform grid search to find the best combination of these two hyperparameters on the developement set.

	MaSciP			**i2b2-2010**		
	Train	Dev	Test	Train	Dev	Test
Number of sentences	1,901	109	158	13,868	2,447	27,625
Number of tokens	61,750	4,158	4,585	129,087	20,454	267,249
Number of mentions	18,874	1,190	1,259	14,376	2,143	31,161
Number of entity types	21	20	21	3	3	3

Table 2: The descriptive statistics of the data sets.

Model	Method	MaSciP				i2b2-2010				Δ
		S	M	L	F	S	M	L	F	
Recurrent	No augmentation	53.0± 3.2	63.0± 0.6	70.3± 0.8	76.4± 0.4	17.1± 2.0	43.3± 1.2	54.1± 0.6	81.1± 0.2	
	Label-wise token rep.	59.7± 0.6	65.5± 0.6	71.4± 0.4	76.3± 0.8	26.7± 0.8	44.3± 0.8	54.5± 0.8	81.0± 0.2	2.6
	Synonym replacement	60.1± 0.5	65.4± 0.4	70.8± 0.6	76.7± 0.8	25.9± 0.5	44.1± 0.5	54.4± 1.5	81.0± 0.3	2.5
	Mention replacement	60.6± 0.6	65.4± 0.4	71.9± 0.5	76.0± 0.8	25.9± 0.7	45.5± 0.4	55.0± 0.2	81.4± 0.2	2.9
	Shuffle within segments	58.8± 0.7	64.6± 0.4	70.5± 0.8	77.1± 0.3	25.2± 0.5	44.4± 0.6	53.5± 0.9	80.6± 0.3	2.0
	All	60.8± 1.3	67.0± 0.8	72.1± 0.7	76.6± 0.4	26.9± 0.7	45.4± 0.6	54.6± 0.9	81.5± 0.2	3.3
Transformer	No augmentation	68.1± 0.6	72.7± 0.3	77.3± 0.5	79.8± 0.7	35.1± 1.1	62.7± 1.5	70.2± 0.3	87.8± 0.2	
	Label-wise token rep.	70.0± 0.8	72.8± 0.2	76.0± 0.6	80.2± 0.6	39.3± 1.7	64.8± 1.3	71.2± 0.4	87.5± 0.2	1.0
	Synonym replacement	70.6± 1.2	73.9± 0.1	76.8± 0.4	79.7± 0.5	42.3± 1.3	65.3± 0.3	70.5± 2.3	87.7± 0.4	1.6
	Mention replacement	70.5± 0.8	73.3± 0.4	76.7± 0.7	80.0± 0.3	40.1± 2.5	64.2± 1.2	70.8± 0.7	87.8± 0.2	1.2
	Shuffle within segments	70.5± 0.4	73.1± 0.6	76.7± 0.3	80.3± 0.5	39.4± 1.6	63.9± 1.4	71.2± 1.2	87.7± 0.2	1.1
	All	71.2± 0.8	73.1± 0.6	76.9± 0.4	80.5± 0.4	41.5± 0.9	65.2± 0.3	72.3± 1.3	87.2± 0.3	1.8

Table 3: Evaluation results in terms of span-level F1 score. **S**mall set contains 50 training instances; **M**edium contains 150 instances; **L**arge contains 500 instances; **F**ull uses the complete training set. We repeat all experiments five times with different random seeds. Mean values and standard deviations are reported. Δ column shows the averaged improvement due to data augmentation. underline: the result is significantly better than the baseline model without data augmentation (paired student's t-test, p: 0.05).

4.4 Results

Table 3 provides the evaluation results on the test sets. The first conclusion we can draw is that all data augmentation techniques can improve over the baseline where no augmentation is used, although there is no single clear winner across both recurrent and transformer models. Synonym replacement outperforms other augmentation on average when transformer models are used, whereas mention replacement appears to be most effective for recurrent models.

Second, applying all data augmentation methods together outperforms any single data augmentation on average, although, when the complete training set is used, applying single data augmentation may achieve better results (c.f., MaSciP-Recurrent and i2b2-2010-Transformer). This scenario may reflect a trade-off between diversity and validity of augmented instances (Hou et al., 2018; Xie et al., 2019). On the one hand, applying all data augmentation together may prevent overfitting via producing diverse training instances. This positive effect is especially useful when the training sets are small. On the other hand, it may also increase the risk of altering the ground-truth label, or generating invalid instances. This negative effect may dominate for larger training sets.

Third, data augmentation techniques are more effective when the training sets are small. For example, all data augmentation methods achieve significant improvements when the training set contains only 50 instances. In contrast, when the complete training sets are used, only three augmentation methods achieve significant improvements and some even decrease the performance. This has also been observed in previous work on machine translation tasks (Fadaee et al., 2017).

Last but not least, we notice that previous studies mainly investigate the effectiveness of data augmentation with recurrent models where most of the parameters are learned from scratch. Considering the significant improvements when using pretrained transformer models, we argue that it is important to investigate the effectiveness of techniques also on pretrained models, such as BERT (Devlin et al., 2019), because they are supposed to capture various knowledge via self-supervision learning.

5 Conclusion

We survey previously used data augmentation methods for sentence-level and sentence-pair NLP tasks and adapt them to NER, a token-level task. Through experiments on two domain-specific data sets, we show that simple data augmentation can improve performance even over strong baselines.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *EMNLP-IJCNLP*, pages 3613–3618, Hong Kong, China.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*, pages 632–642, Lisbon, Portugal.

Hengyi Cai, Hongshen Chen, Yonghao Song, Cheng Zhang, Xiaofang Zhao, and Dawei Yin. 2020. Data manipulation: Towards effective instance learning for neural dialogue generation via learning to augment and reweight. In *ACL*, pages 6334–6343, Online.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very deep convolutional networks for text classification. In *EACL*, pages 1107–1116, Valencia, Spain.

Xiang Dai, Sarvnaz Karimi, Ben Hachey, and Cecile Paris. 2020. Cost-effective selection of pretraining data: A case study of pretraining BERT on social media. *arXiv preprint arXiv:2010.01150*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186, Minneapolis, Minnesota.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *ACL*, pages 567–573, Vancouver, Canada.

Annemarie Friedrich, Heike Adel, Federico Tomazic, Johannes Hingerl, Renou Benteau, Anika Marusczyk, and Lukas Lange. 2020. The SOFC-exp corpus and neural approaches to information extraction in the materials science domain. In *ACL*, pages 1255–1268, Online.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *ACL*, pages 5539–5544, Florence, Italy.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *ICASSP*, pages 6645–6649.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *ACL*, pages 8342–8360, Online.

Yutai Hou, Yijia Liu, Wanxiang Che, and Ting Liu. 2018. Sequence-to-sequence data augmentation for dialogue language understanding. In *COLING*, pages 1234–1245, Santa Fe, New Mexico, USA.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *ACL-IJCNLP*, pages 1681–1691, Beijing, China.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *NAACL*, pages 1875–1885, New Orleans, Louisiana.

Sarvnaz Karimi, Alejandro Metke-Jimenez, Madonna Kemp, and Chen Wang. 2015. CADEC: A corpus of adverse drug event annotations. *J Biomed Inform*, 55:73–81.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *NAACL*, pages 452–457, New Orleans, Louisiana.

Gakuto Kurata, Bing Xiang, and Bowen Zhou. 2016. Labeled data generation with encoder-decoder lstm for semantic slot filling. In *INTERSPEECH*, pages 725–729.

Sepideh Mesbah, Jie Yang, Robert-Jan Sips, Manuel Valle Torre, Christoph Lofi, Alessandro Bozzon, and Geert-Jan Houben. 2019. Training data augmentation for detecting adverse drug reactions in user-generated content. In *EMNLP-IJCNLP*, pages 2349–2359, Hong Kong, China.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *ACL*, pages 2339–2352, Online.

Sheshera Mysore, Zachary Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. 2019. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. In *ACL@LAW*, pages 56–64, Florence, Italy.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*, pages 1532–1543, Doha, Qatar.

Jonathan Raiman and John Miller. 2017. Globally normalized reader. In *EMNLP*, pages 1059–1069, Copenhagen, Denmark.

Sebastian Ruder. 2019. *Neural Transfer Learning for Natural Language Processing*. Ph.D. thesis, National University of Ireland, Galway.

Gözde Gül Şahin and Mark Steedman. 2018. Data augmentation via dependency tree morphing for low-resource languages. In *EMNLP*, pages 5004–5009, Brussels, Belgium.

Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2011. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *J. Am. Med. Inform. Assoc.*, 18(5):552–556.

Jason Wang and Luis Perez. 2017. The effectiveness of data augmentation in image classification using deep learning. *CoRR abs/1712.04621*.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In *EMNLP*, pages 2557–2563, Lisbon, Portugal.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *EMNLP*, pages 856–861, Brussels, Belgium.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *EMNLP-IJCNLP*, pages 6382–6388, Hong Kong, China.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2018. Conditional bert contextual augmentation. *CoRR abs/1812.06705*.

Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. 2019. Generalized data augmentation for low-resource translation. In *ACL*, pages 5786–5796, Florence, Italy.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. 2019. Unsupervised data augmentation. *CoRR abs/1904.12848*.

Kang Min Yoo, Youhyun Shin, and Sang-goo Lee. 2019. Data augmentation for spoken language understanding via joint variational generation. In *AAAI*, Honolulu, Hawaii.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. QANet: Combining local convolution with global self-attention for reading comprehension. In *ICLR*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*, pages 649–657.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *NAACL*, pages 15–20, New Orleans, Louisiana.