Transfer Learning from Pre-trained BERT for Pronoun Resolution

Xingce Bao[*]
School of Engineering, EPFL
Switzerland
xingce.bao@epfl.ch

Qianqian Qiao[*]
School of Engineering, EPFL
Switzerland
qianqian.qiao@epfl.ch

Abstract

The paper describes the submission of the team ”We used bert!” to the shared task Gendered Pronoun Resolution (Pair pronouns to their correct entities). Our final submission model based on the fine-tuned BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018) ranks 14th among 838 teams with a multi-class logarithmic loss of 0.208. In this work, contribution of transfer learning technique to pronoun resolution systems is investigated and the gender bias contained in classification models is evaluated.

1 Introduction

The shared task Gendered Pronoun Resolution aims to classify the pronoun resolution in the sentences, hereby to find the true name referred by a given pronoun, such as ***she*** in:

*In May, **Fujisawa** joined Mari Motohashi's rink as the team's skip, moving back from Karuizawa to Kitami where **she** had spent her junior days.*

This task for pronoun resolution closely relates to the traditional coreference resolution task in natural language processing. Many works (Wiseman et al., 2016; Clark and Manning, 2016; Lee et al., 2017) related to coreference resolution have been published recently and all of them are evaluated with CoNLL-2012 shared task dataset (Pradhan et al., 2012). However, simply pursuing the best score over the entire dataset may cause the neglect of the model performance gap between the two genders.

To explore the existence of gender bias in such tasks, researchers from Google built and released GAP (Gendered Ambiguous Pronouns) (Webster et al., 2018), a human-labeled corpus of 8908 ambiguous pronoun-name pairs derived from Wikipedia with balanced gender pronouns. It has been shown that most of the recent representative coreference systems struggled on GAP dataset with a overall mediocre performance and a large performance gap between genders. This may be due to both unbalanced training dataset used by these coreference systems or the design of the systems. Up to now, detecting and eliminating gender bias in such systems still remains a challenge.

In this paper, we explore transfer learning from pre-trained models to improve the performance of tasks with limited data. Various efficient approaches to reuse the knowledge from pre-trained BERT on this shared task are proposed and compared. The final system significantly outperforms the off-the-shelf resolvers, with a balanced prediction performance for two genders. Moreover, gender bias in word and sentence level embeddings is studied with a scientific statistical experiment on Caliskan dataset (Caliskan et al., 2017).

2 Data

This shared task is based on GAP dataset including:

- Test 4,000 pairs: used for official evaluation
- Development 4,000 pairs: used for model development
- Validation 908 pairs: used for parameter tuning

In the first stage, we use part of the released data on Google GAP Github repository, which includes 2000 development pairs, 2000 test pairs, and 454 validation pairs.[1] We refer the test pairs as training

[*] Both authors contributed equally in this work.

[1] The testing data from the Kaggle website is the development data in the GAP github repository. So we use the development pairs to evaluate our model, and the test pairs to train in order to conform the Kaggle competition rule.

Proceedings of the 1st Workshop on Gender Bias in Natural Language Processing, pages 82–88
Florence, Italy, August 2, 2019. ©2019 Association for Computational Linguistics

data, the development pairs as testing data and the validation pairs as validation data. Each sample contains a sentence and three mentions, A, B and pronoun. Each pronoun has been labeled as A, B, or NEITHER. Submissions are evaluated using the multi-class logarithmic loss.

Table 1 shows the frequency of the different types of pronouns in the dataset. The number of masculine pronouns and feminine pronouns are strictly equal.

Pronoun type	Training	Test	Validation
he	348	373	93
him	96	98	26
his	556	529	108
her	603	572	140
hers	1	0	0
she	396	428	87
masculine	1000	1000	227
feminine	1000	1000	227

Table 1: Pronoun gender frequency

3 Data Preparation

We introduced the procedure for processing the data before training in detail in this section.

3.1 Data Preprocessing

Data preprocessing can be summarized into the following steps:

BERT embeddings generation: We use pre-trained bert-large-uncased model to obtain contextual embeddings as features. This part is implemented with the bert-as-service library based on Tensorflow (Xiao, 2018).

Dimension reduction: The dimension reduction for the original BERT contextual embeddings is performed to mitigate the overfitting problems. This approach is inspired by the Algorithm 2 (PPA-PCA-PPA) proposed in Raunak (2017).

For large scale vectors with dimension of 1024, instead of directly using PCA (principal component analysis), we train a linear autoencoder to approximate the linear PCA procedure. Namely, we train the autoencoder by minimizing the loss:

$$L(X, W_1, W_2) = ||X - W_2 W_1 X||_2^2, \quad (1)$$

where X is the contextual embedding. W_1 and W_2 are $m \times n$ and $n \times m$ matrices to project vectors to lower dimensional space and recover from lower dimensional space, respectively ($m < n$). Hence, the PCA part in the original algorithm is performed by computing $W_1 X$, and the PPA part in the original algorithm is performed by computing $X - W_2 W_1 X$.

Here the PPA procedures remove the first 4 principal components. The PCA procedure maps 1024 dimension vectors to 256 dimension vectors.

Processing mention: A mention in the data (A, B or the pronoun) can be a single word or multiple words. Also, since BERT is based on the word piece model (Wu et al., 2016), a word may be cut into multiple word pieces after the BERT tokenization. We define the mention index as the index for the tokenized word piece list which corresponds to the original mention.

The vectors in the BERT contextual embeddings which correspond to the mention index are extracted. Meanwhile, vectors of mentions are the mean value of all the vectors which correspond to the mention. We call this mention vector.

Find names: All names in the sentences except A and B are extracted with the named entity recognition tool. After that, their mention indices are found by the same procedure in the previous step. We call these indices neither mention index. Stanford Named Entity Tagger is used for finding the names in the sentences in this step (Finkel et al., 2005).

An example of tokenization and mention index is shown in table 2.

Sentence: When asked in a 2010 interview with The Mirror what **her** favourite scenes were, **Beverley Callard** replied, “when Jim beat up **Liz**.
Names Except A and B: Jim
Tokens: [’when’, ’asked’, ’in’, ’a’, ’2010’, ’interview’, ’with’, ’the’, ’mirror’, ’what’, **’her’**, ’favourite’, ’scenes’, ’were’, ’,’, **’beverley’, ’call’, ’##ard’**, ’replied’, ’,’, ’“’, ’“’, ’when’, ’jim’, ’beat’, ’up’, **’liz’**, ’.’]
Mention A: Beverley Callard **Mention B: Liz** **Mention Pronoun: her** **Mention Neither: Jim**
A Mention Index: 15,16,17 **B Mention Index: 26** **Pronoun Mention Index: 10** **Neither Mention Index: 23**

Table 2: An example of tokenization and mention index

3.2 Data Augmentation

We replace the originally referred mention by a different random mention in the sentence, then change the label to neither. This creates 1445 samples labeled neither from training data. Original training data together with augmented neither data make up the augmented training set.

4 Architecture

We mainly explored two sub-categories of models as shown in figure 1. One category is based on fine-tuned BERT with different top layers. For this category, Back-propagation is done to both top layers and the pre-trained BERT model. Another idea is to use BERT as a feature extractor. Different from fine-tuned BERT, models in the second category do not back propagate to BERT weights during training. All of these base models contribute to our final model.[2]

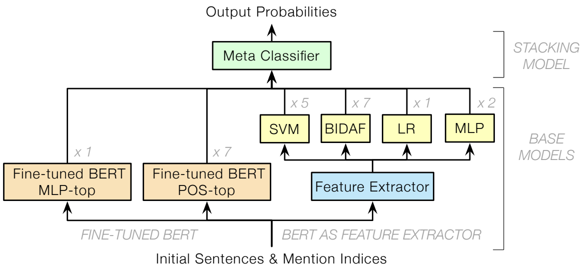


Figure 1: Structure of the final system. It contains 23 base models with different structures, different embedding dimensions and data whether augmented.

4.1 Fine-tuned BERT

We propose two different kinds of top layers to fine-tune BERT model on GAP task and implemented with PyTorch Pretrained BERT library(Hugging-Face, 2018). The first kind of top layer shown in figure 2 is called MLP-top. It extracts and aggregates vectors for all mentions by concatenation, which are then fed into a multiple layer neural network.

The second kind of top layer first map the output of BERT into a scalar by a linear layer whose output size is 1. Then we extract the value corresponding to the mention index and feed it into a softmax layer for a 3-class-probability-output. We call this Positional-top which is illustrated in Figure 3.[3]

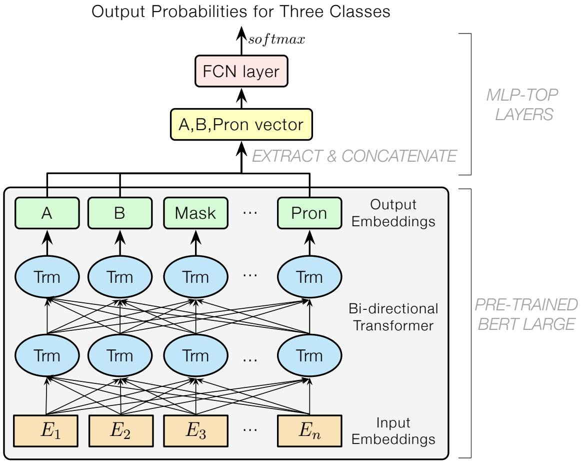


Figure 2: Fine-tuned BERT with MLP-top layer

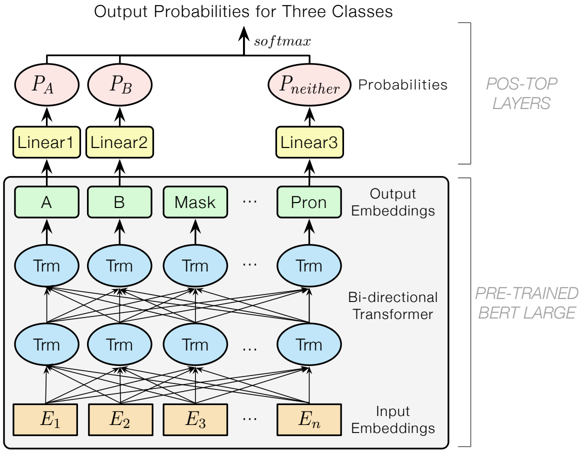


Figure 3: Fine-tuned BERT with Positional-top layer. Linear layers for A,B and Pronoun are with the same parameter.

4.2 BERT as Feature Extractor

When BERT is used as a feature extractor, the contextual embeddings and the mention vectors prepared are passed to the subsequent classifier. Here we use SVM (support vector machine) and BIDAF (bi-directional attention flow layer) (Seo et al., 2017) as classifiers.

SVM: We denote the mention vector of A, B and pronoun as $\mathbf{h_A}$, $\mathbf{h_B}$ and $\mathbf{h_{pron}}$. The vector:

$$[\mathbf{h_A}, \mathbf{h_B}, \mathbf{h_{pron}}, \mathbf{h_A} \odot \mathbf{h_{pron}}, \mathbf{h_B} \odot \mathbf{h_{pron}}] \quad (2)$$

is fed as the input of the SVM, where the $\odot$ means point-wise product. The multiclass support is handled according to a one-vs-one scheme. The SVM

[2] Due to the space limit, we do not explain all the base models that we use to produce the final ensemble model in detail. The models in the following description are only efficient and representative base models. For a comprehensive list of the base models we use, please check: https://github.com/bxclib2/kaggle_gender_coref/

[3] Both figure 2 and figure 3 show the mentions which contain only a single word-piece after tokenization. If one mention contains multiple word-pieces, the mean of the multiple positions in BERT output layer should be computed in order to generate a tensor with desired size to be fed into the top layer.

classifier is implemented with Scikit-Learn library (Pedregosa et al., 2011).

BIDAF: BERT contextual embeddings and the pronoun mention vectors are passed to the bidirectional attention flow layer as the context and the query, respectively. We use the original embedding extracted from BERT large with embedding dimension of 1024 here. Then a two-layer point-wise fully-connected neural network is connected to map the output embedding vectors to scalars. The fully-connected layer has 64 hidden units with ELU as activation function (Djork-Arn Clevert, 2016). Finally, the scalars corresponding to the A, the B and the neither are fed into a softmax layer to generate 3-class probabilities.

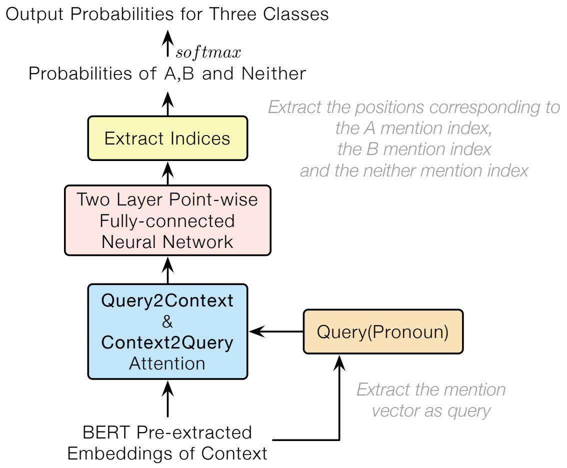

Output Probabilities for Three Classes
$softmax$
Probabilities of A,B and Neither
Extract Indices
Extract the positions corresponding to the A mention index, the B mention index and the neither mention index
Two Layer Point-wise Fully-connected Neural Network
Query2Context & Context2Query Attention
Query(Pronoun)
Extract the mention vector as query
BERT Pre-extracted Embeddings of Context

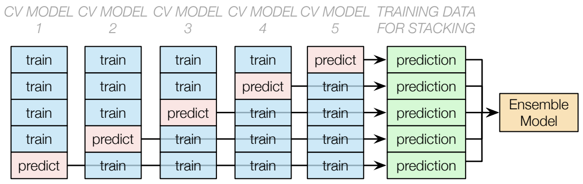
Figure 4: Structure of BIDAF network

The top layer of BIDAF network works similarly to the positional head of the fine-tuned BERT. However, there are two major differences: the positional head of the fine-tuned BERT uses only a linear layer to map the embeddings to scalars, while the BIDAF network uses a two-layer neural network with the ELU activation layer. Also, the output of BIDAF is from the positions corresponding to the A, the B and the neither mention respectively, while the BERT positional head extracts the scalars corresponding to the A, the B and the pronoun mention respectively.

4.3 Model Ensemble

Ensemble learning greatly improves the results compared to single models. Stacking method is used for ensemble. During ensemble, several base classifiers are trained to make preliminary predictions, and a meta classifier is used to make a final prediction based on these predictions.

In order to reduce the data leakage, 5-fold cross validation is performed when building the training data for the meta classifier from the original training data. In other words, we avoid the base classifiers and meta classifier to be trained with the same fold of data (Beaudon, 2016). For each training time 4-fold of data is used to train, and the resulting model predicts the remaining one fold of data to build one fold of training data for the meta classifier, as shown in figure 5. Here we use the logistic regression as the meta classifier.



CV MODEL 1	CV MODEL 2	CV MODEL 3	CV MODEL 4	CV MODEL 5	TRAINING DATA FOR STACKING
train	train	train	train	predict	prediction
train	train	train	predict	train	prediction
train	train	predict	train	train	prediction
train	predict	train	train	train	prediction
predict	train	train	train	train	prediction

Figure 5: 5-Fold cross validation for stacking

5 Experiment

In this section, we present the result of different classifiers to the shared task.

5.1 Experiment setting

For SVM, C equals to 5.0 and the kernel function is the RBF function. The SVM is trained both with the original 1024 dimension mention vectors and the 256 dimension-reduced mention vectors respectively for comparison.

The BIDAF network is trained for 50 epoches with a batch size of 25. We use the Adam optimizer with a learning rate of 1e-3 for training. For each fully-connected layer in BIDAF, a dropout with probability 0.7 is performed. It is trained both with the original training set and the augmented training set for comparison. This training process takes about 10 minutes with the GTX 1070 GPU.

The fine-tuned BERT models are trained with the Adam optimizer with a learning rate of 2e-5. All the dropout layers in the original BERT model are set to a dropout rate of 0.15. Models are trained for 1 epoch with a batch size of 16. Note that it is not possible to fit 16 training sentences at one time due to the limited GPU memory. Hence, gradient accumulation trick is used. Every time we fit 2 training sentences and we accumulate the gradient for 8 times. This fine-tuning process takes about 10 minutes with the Tesla K80 GPU.

The meta classifier is the logistic regression with l_2 regularization of the regularization constant C which equals to 0.5.

5.2 Evaluation

The results are shown in table 3. The masculine data loss and feminine data loss are shown respectively in order to show the gender bias. We compute the model loss for testing data (stage 1) and the loss caused by the masculine part and the feminine part in stage 1 testing data. We also submit our base model results after the competition finishes in order to get the private testing data (stage 2) loss.

	M	**F**	**T**	**PT**
SVM 256	0.516	0.495	0.506	0.395
SVM 1024	0.619	0.574	0.596	0.475
BIDAF	0.490	0.498	0.494	0.364
BIDAF-aug	0.550	0.579	0.565	0.422
BERT-pos	0.376	0.377	0.377	**0.280**
BERT-mlp	**0.360**	**0.365**	**0.362**	0.351
Ensemble	**0.325**	**0.337**	**0.331**	**0.208**

Table 3: Evaluation results (multi-class logarithmic loss) for models. SVM 256: SVM trained with the mention vector after dimension reduction. SVM 1024: SVM trained with the original 1024 dimension mention vector. BIDAF: BIDAF trained with the original training set. BIDAF-aug: BIDAF trained with the augmented training set. BERT-pos: Fine-tuned BERT with the Positional-top. BERT-mlp: Fine-tuned BERT with the MLP-top. **M**asculine, **F**eminine, **T**esting data and **P**rivate **T**esting data results are shown respectively. Bold indicates the best performance.

We derive the following conclusions:

- The dimension reduction greatly enhances the result of SVM which reduces about 0.1 multi-class logarithmic loss. The SVM 1024 has a loss of 0.184 and 0.597 with respect to training and testing data, while the SVM 256 has a loss of 0.250 and 0.505. Both SVM model overfit a lot, while the dimension reduction of BERT contextual embeddings efficiently mitigate overfitting, which bridges the performance gap between training data and testing data.

- The BIDAF model performs worse when trained with the augmented training set than the original training set, due to the distribution mismatching caused by data augmentation that, the portion of the neither data is larger in the training set than in the testing set.

- Both two fine-tuned BERT models achieve much more competitive results compared to Bert as Feature Extractor models.[4]

- The ensemble learning with logistic regression greatly enhances the overall classification result.

Although the data augmentation does not improve the BIDAF model directly, it still helps to make more accurate predictions of the neither class in the ensemble model. The BIDAF-aug and the BIDAF reach the loss of 0.982 and 1.095, respectively. In the testing data (stage 1), the respective accuracy of A, B and neither class is 89.8%, 89.5% and 73.1%, indicating that predicting the neither class correctly is much harder than predicting A and B. We can observe that it is easier for the model to choose an answer as A or B than to predict as no reference.

We also evaluate our system F1 score with stage 1 testing dataset to compare to the off-the-shelf resolvers in table 4:

	M	**F**	**B**	**O**
Wiseman et al.	68.4	59.9	0.88	64.2
Lee et al.	67.2	62.2	0.92	64.7
BERT-pos	**86.8**	**86.1**	0.99	**86.5**
BERT-mlp	86.3	85.9	**1.00**	86.1
Our ensemble	**88.1**	**87.9**	**1.00**	**88.0**

Table 4: Comparison to off-the-shelf resolvers, split by **M**asculine and **F**eminine (**B**ias shows F/M), and **O**verall. Bold indicates the best performance.

6 Gender Bias in the Embeddings

To further demonstrate the presence or absence of gender bias in embeddings, we use both the Word Embedding Association Test (WEAT) (Caliskan et al., 2017) and Sentence Embedding Association Test (SEAT) (May et al., 2019) to measure it. As fine-tuned BERT large models with Positional-top contribute a lot to our final ensemble model, we only focus on this category of models in this section.

[4] Here the experiment shows that the MLP-top is slightly better than the Positional-top. However, the Positional-top is more stable with different random seeds. Also it is obvious that the MLP-top performs worse than the Positional-top in the private testing data.

6.1 WEAT & SEAT

For both word-level test and sentence level test, let X and Y be two sets of target concept word or sentence embeddings, and let A and B be two sets of attribute word embeddings. The test statistic is the difference between sums of similarities of the respective attributes over target concepts, which can be calculated as:

$$s(X,Y,A,B) = \sum_{x \in X} s(x,A,B) - \sum_{y \in Y} s(y,A,B), \quad (3)$$

where:

$$\begin{aligned} s(w,A,B) = &\text{mean}_{a \in A}\cos(w,a) - \\ &\text{mean}_{b \in B}\cos(w,b), \end{aligned} \quad (4)$$

the p-values on $s(X,Y,A,B)$ is used to compute the significance between (A,B) and (X,Y),

$$p = \Pr[s(X_i,Y_i,A,B) > s(X,Y,A,B)], \quad (5)$$

where X_i and Y_i are of equal size. Also the effect size d is used to measure the magnitude of associations:

$$d = \frac{\text{mean}_{x \in X} s(x,A,B) - \text{mean}_{y \in Y} s(y,A,B)}{\text{std\_dev}_{w \in X \cup Y} s(w,A,B)} \quad (6)$$

6.2 Experiments and Results

We apply WEAT and SEAT on Caliskan Test of male/female names with career and family, which corresponds to past social psychology studies.

Method	GloVe	ELMo	BERT	F-BERT
WEAT	1.81*	−0.45	0.21	0.38
SEAT	1.74*	−0.38	0.08	0.07

Table 5: Effect sizes for male/female names with career/family task with word and sentence level embeddings. *: significant at 0.01. F-BERT indicates Fine-tuned BERT.

Table 5 shows the result of WEAT and SEAT. Sentence vectors are aggregated by taking the mean value of all word vectors in the sentences for GloVe (Pennington et al., 2014), ELMo (Peters et al., 2018), BERT and Fine-tuned BERT.[5] With p-values lower than 0.01, embeddings by GloVe on both word level and sentence level show significant gender bias, indicating that women are associated with family while men are associated with career.

However, p-values of all contextual embeddings including ELMo, BERT and Fined-tuned BERT are larger than 0.05, which suggests that there is no evidence suggesting existence of gender bias in these embeddings. One possible explanation is that, by training contextual word embeddings, a single word is usually represented differently in different sentences, resulting in more flexible word representations focusing on single context within a sentence rather than the overall word frequency distribution.

7 Conclusion and Future Work

We propose a transfer-learning-based solution for pronoun resolution. The proposed solution leads to gender balance in both word embeddings and overall predictions. It greatly improves the prediction accuracy of this task by 23.3% F1 against the off-the-shelf solutions proposed by Lee et al. (2017) on the widely studied Google GAP dataset. Meanwhile, among several single models in our ensemble solution, BERT-mlp and BERT-pos model highly outperform others in the experiments. Overall this work shows the efficacy of employing BERT in downstream natural language processing classification tasks.

In the future, we would like to investigate various transfer structures on the top of pre-trained BERT, especially for the sake of enhancing the stability of the fine-tune process. We observe in our experiments that the performance of fine-tune models based on BERT strongly depends on initial random state, thus, further research on building more robust models is indispensable.

Acknowledgments

We thank for Wanhao Zhou, Siyuan Li, Fengyu Cai for useful discussions. We also thank Ruocheng Han, Tian Guo, Xi Fan and Jirou Feng for feedback and writing help.

References

Romain Beaudon. 2016. Cross validation strategy when blending/stacking. https://www.kaggle.com/general/18793.

[5] Here we use a different method to aggregate sentence vector for BERT, comparing to the cited paper which uses [CLS] vector as sentence vector for better comparison.

Aylin Caliskan, Joanna Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356:183–186.

Kevin Clark and Christopher D. Manning. 2016. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653, Berlin, Germany. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *Computing Research Repository*, arXiv:1810.04805. Version 1.

Sepp Hochreiter Djork-Arn Clevert, Thomas Unterthiner. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *International Conference on Learning Representations*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370, Ann Arbor, Michigan. Association for Computational Linguistics.

Hugging-Face. 2018. pytorch-pretrained-bert. `https://github.com/huggingface/pytorch-pretrained-BERT`.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. *Computing Research Repository*, arXiv:1903.10561. Version 1.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Vikas Raunak. 2017. Simple and effective dimensionality reduction for word embeddings. In *Proceedings of the workshop on the learning with limited labeled data, NIPS'2017*, Long Beach, CA, USA.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations*.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, 6:605–617.

Sam Wiseman, Alexander M. Rush, and Stuart M. Shieber. 2016. Learning global features for coreference resolution. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 994–1004, San Diego, California. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *Computing Research Repository*, arXiv:1609.08144. Version 2.

Han Xiao. 2018. bert-as-service. `https://github.com/hanxiao/bert-as-service`.