Gendered Pronoun Resolution

Mustafa Abdool
Stanford University
moose878@stanford.edu

Sarah Egler
Stanford University
segler@stanford.edu

Abstract

Most state-of-the-art coreference models are trained and evaluated on gender imbalanced datasets such as the widely used OntoNotes [13]. Such datasets include far more examples of male pronouns than female pronouns, yielding models whose performance differ across gender. We seek to build a coreference resolution system that can perform well regardless of pronoun gender. We use a balanced corpus of gender ambiguous pronouns known as GAP [1]. We perform a gold-two-mention version of the task, in which the pronoun is given along with two candidate mentions. We train a basic neural network with linear layers/ReLU using simple token distance features as a baseline. We then show several improvements on this baseline using various state of the art techniques described in literature such as word and character embeddings, hand-engineered features [5], recurrent neural networks [2] and transformer models [7]. We achieve best overall performance with hand-crafted features and transformers, and find that training on a gender-balanced dataset mitigates gender-biased performance on the task.

1 Introduction

Coreference resolution is the task of identifying all mentions within a text that refer to the same entity and has been a long standing challenge in Natural Language Processing. Recently, coreference resolution was proposed as a viable alternative to the Turing test [11]. In recent years, the state of the art co-reference resolution systems have transitioned from heuristic algorithms [6], to mention-pair ranking models with hand engineered features [5] to a fully end-to-end deep learning solution [2].

However, most state-of-the-art coreference models are trained and evaluated on gender imbalanced datasets such as the widely used OntoNotes [13]. The GAP dataset, a gender-balanced corpus of ambiguous pronouns, was created to address this. State-of-the-art models perform quite poorly when tested on the GAP dataset, and fail to even outperform simple baselines based on parallelism and syntactic cues [1].

We perform a gold-two-mention version of coreference resolution on the GAP dataset in which we have access to the target pronoun and two candidate mentions and output a probability that each candidate, or neither, corefers with the target pronoun. Our main approaches involve taking inspiration from various state of the art systems for NLP tasks and applying them to the GAP dataset with appropriate modifications. These approaches include using hand-engineered language features [5], word embeddings and character embeddings of key words [10], recurrent neural networks [2], transformer models [7] and a combination thereof. Specifically, since the dataset is relatively small, many of our approaches involve using transfer learning of pre-trained embedding models such as GloVe [12], a gender-neutral variant of GloVe [3], and BERT [8].

2 Related Work

The creators of the GAP dataset discuss the performance of several off-the-shelf resolvers on the GAP validation set as well as some simple baselines such as features based on parallelism cues that we use

Table 1: Sample of Hand-Crafted S ntacticyFeatures

Feature
Count of pronouns in the text
Count of pronouns between target pronoun and candidate
Position of candidate/target pronoun in sentence
Position of sentence containing candidate/target pronoun in document
Syntactic Dependency of candidate/target pronoun
Number of times candidate appears in text

to benchmark our own results [1]. Our approach is inspired by the two state of the art systems: (1) the end-to-end neural network resolver [2] that the GAP authors found to perform best on the GAP development set, and (2) the Clark and Manning neural network resolver with hand crafted features [5] that the GAP authors noted performed best on the OntoNotes test set. The *gold-two-mention task*) in which we have access to the two candidate spans is slightly different as compared to the entity clustering problem in these models, but there is still much we can draw from these models.

Clark and Manning detail a Mention-Pair Encoder which takes as input hand engineered features from the text and outputs a Mention-Pair Representation. The features used in this approach include embedding features such as the head word and surrounding words of the mention, mention features such as type and position of the mention, document genre, distance features, speaker, and string match features [5]. We use several of these features in our own approach.

Lee et al. provide the first end-to-end approach to coreference resolution, using a bidirectional LSTM over word and character level embeddings to generate span representations with attention to find the head word. Mention scores are calculated for each span, along with antecedent scores for each pair of spans, which together form the basis of a coreference score [2]. As we already have access to candidate mentions and the pronoun, we need not compute mention scores but draw from the LSTM and antecedent scoring approach.

3 Approach

3.1 Baseline

We consider two baseline models. The first model was simply using the off-the-shelf HuggingFace Neural Coreference model [4] which detects clusters of coreference given text. We had to manipulate the output of this model slightly to frame it as our specific *gold-two-mention* task which involved checking if the target pronoun and candidate were in the same cluster. The second baseline model we used was a two layer neural network trained with basic features, h_0, such as the total length of the text, total number of words in the text and the distance between the target pronoun and the candidate. This model was implemented from scratch in Pytorch as follows, outputting a probability, p_i, for each candidate or neither:

$$h_1 = ReLU(Wh_0 + b) \tag{1}$$

$$h_2 = ReLU(Wh_1 + b) \tag{2}$$

$$logits = ReLU(Wh_2 + b) \tag{3}$$

$$p_{candidate} = Softmax(logits) \tag{4}$$

3.2 Approach 1: Hand-Crafted Features

For our first approach, we used feature engineering to derive a variety of linguistic-based features as in [5]. See table 1 for examples of some of the key features included. This model was implemented from scratch in Pytorch.

3.3 Approach 2: Word Embeddings

We use a gender neutral variant of GloVe word embeddings, GN-GloVe [3]. These word embeddings allow for gender information in certain dimensions of word embeddings, while preventing the presence of gender in other dimensions. GN-GloVe contains approximately 300,000 of these 300-dimensional embeddings. We compare the GN-GloVe embeddings with 300-dimensional GloVe embeddings to see if this affects gender bias on our task. Word embeddings are computed for the target pronoun, the candidate, the head word of the target and candidate along with the two preceding and following words [5]. We then concatenate these embeddings at layer 2 of the hand-crafted feature model described in Approach 1.

3.4 Approach 3: Character Embeddings

This approach is inspired by the approach of using character embeddings for language modelling tasks [10]. We believe that character embeddings may help provide information around syntactic structure, such as with the possessive form of mentions as in *Abi's* or with punctuation that signals complex text structure such as commas and parentheses. Firstly, we exract key words from the text (target pronoun, head word of both candidates, head word of target pronoun) similar to the Word Embeddings approach. We then lookup the character embedding for each word and feed those through a character based convolution to get an embedding representation for each word. This process is described by the equations below.

$$\mathbf{w} = WordExtractor(text) \tag{5}$$

$$\mathbf{w_{embed}} = CharEmbed(\mathbf{w}) \tag{6}$$

$$\mathbf{w_{conv}} = Conv1D(\mathbf{w_{embed}}) \tag{7}$$

Lastly, we feed $\mathbf{w_{embed}}$ through a highway network [9] as they have been shown to empirically improve results for a variety of language modelling tasks.

3.5 Approach 4: Bidirectional LSTM with Attention

Our third approach is inspired by the end-to-end coreference resolver [2] that the authors of the GAP paper found to perform best on the GAP dataset [1]. The idea is that the two key components to learning the task of coreference resolution are the syntactic structures within a span and the context surrounding the span.

We first compute the span candidate and pronoun using the spaCy span API. We feed our word representations, which are the gender neutral word embeddings along with the character level embeddings (computed using a one-dimensional CNN) into a bi-directional LSTM with attention to the pronoun hidden state. We compute a coreference score for a candidate span, g_i, with the pronoun span, g_j, according to the following equation:

$$s(i,j) = w \cdot \texttt{FFNN}(g_i, g_j, g_i \circ g_j, \phi(i,j)) \tag{8}$$

where the FFNN is a nonlinear input to output mapping of the spans, $\circ$ represents elementwise multiplication, and $\phi(i,j)$ are additional context relevant features as described in our hand-crafted features.

3.6 Approach 5: Transformer models / BERT

Recently, transformer models have shown much promise in various NLP tasks even achieving state of the art results in machine translation [8]. We use a pre-trained version of Google AI's Bidirectional Encoder Representations from Transformers (BERT) model to embed the entire text. Then, we extract embedding representations for key words (the target pronoun and two candidate references) for fine-tuning. We combine these representations with our own hand engineered distance and parallelism/syntactic dependency features in a higher layer to achieve our best results. The equations below describe this process.

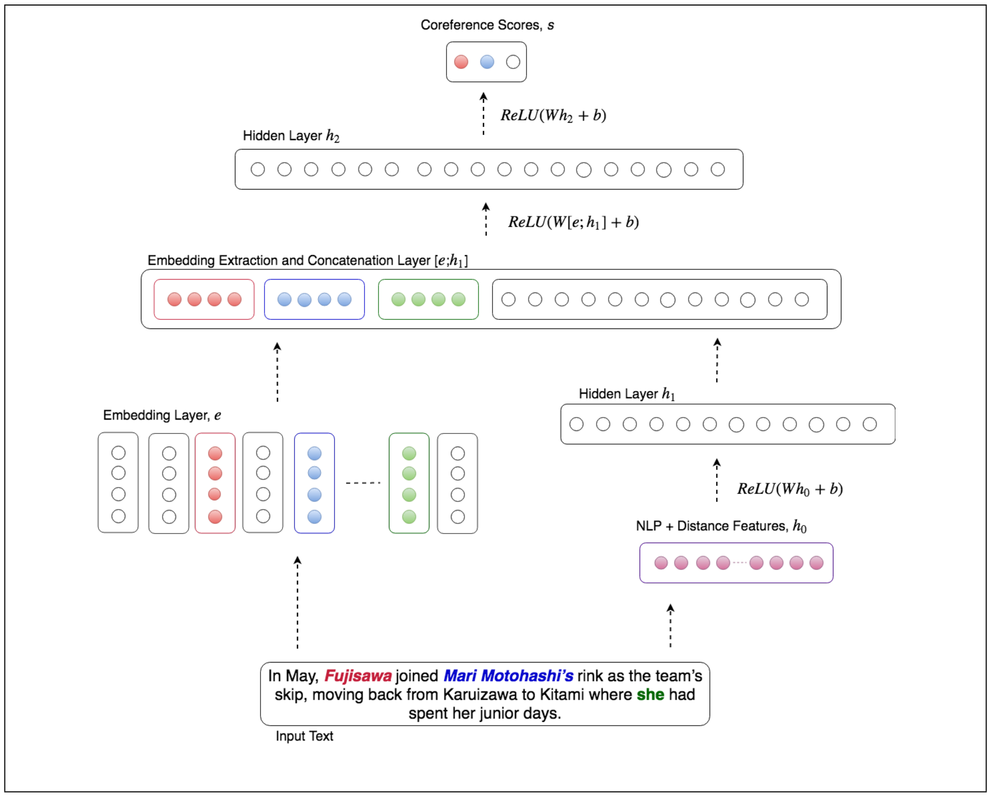
Coreference Scores, s
ReLU(Wh_2 + b)
Hidden Layer h_2
ReLU(W[e; h_1] + b)
Embedding Extraction and Concatenation Layer [e;h_1]
Hidden Layer h_1
Embedding Layer, e
ReLU(Wh_0 + b)
NLP + Distance Features, h_0
In May, Fujisawa joined Mari Motohashi's rink as the team's skip, moving back from Karuizawa to Kitami where she had spent her junior days.
Input Text

Figure 1: Architecture used in Approach 2. The input is comprised of pretrained word embeddings and hand-engineered syntactic features and the output layer contains scores for whether candidate A, B, or neither refers to the same entity as the pronoun.

$$\mathbf{b} = BertModel(text) \tag{9}$$

$$\mathbf{b_{targetwords}} = \left[\mathbf{b}_{o_{target}}; \mathbf{b}_{o_1}; \mathbf{b}_{o_2}\right] \tag{10}$$

$$\mathbf{s_{output}} = \texttt{FFNN}(b_{targetwords}, h_1) \tag{11}$$

Here, o_{target}, o_1, o_2 are the offsets of the tokens corresponding to the target pronoun and two candidate words in the output of the BERT model. These extracted outputs are concatenated with the hidden layer of the network that processes the hand crafted distance and parallelism features and then fed through a final network to produce the output logits.

4 Experiments

4.1 Data

As mentioned previously, the main dataset we use for training/evaluation is the GAP dataset [1]. The dataset has 4000 examples for training, 4,000 examples for test and 908 examples for validation. See table 2 for a full description of all columns in the dataset.

4.2 Evaluation Method

For each example, we predict the probability that candidate A corefers with the pronoun, candidate B corefers with the pronoun, or neither candidate A nor candidate B corefers with the pronoun. We evaluate our performance using the multiclass logarithmic loss defined below, where N is the number of examples, and $M = 3$ according to the three classes (candidate A, candidate B, neither):

Table 2: GAP Dataset Overview

Column	Header	Description
1	ID	Unique identifier for an example (two pairs)
2	Text	Text containing the ambiguous pronoun and two candidate names.
3	Pronoun	The pronoun, text
4	Pronoun-offset	Character offset of Pronoun in Column 2 (Text)
5	A	The first name, text
6	A-offset	Character offset of A in Column 2 (Text)
7	A-coref	Whether A corefers with the pronoun, TRUE or FALSE
8	B	The second name, text
9	B-offset	Character offset of B in Column 2 (Text)
10	A-coref	Whether B corefers with the pronoun, TRUE or FALSE
11	URL ‘	The URL of the source Wikipedia Page

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} log(p_{ij}) \tag{12}$$

Additionally, we take the argmax over the probabilities for each class to output a binary prediction as to whether a candidate corefers with the pronoun or not and evaluate this output with the metric of F1 score.

In order to measure gender bias, in addition to an overall F1 score **O**, we also use separate F1 scores for the masculine, **M**, and feminine, **F**, examples and take a feminine-to-masculine F1 score ratio, **F/M**, as a measure of bias as done in the original GAP paper [1].

This is a Kaggle competition and so we evaluate the relative performance of our models against the leaderboard which uses the multiclass logarithmic loss.

4.3 Experiment Details

4.3.1 General Parameters

All the models trained below used the Adam optimizer and were initialized with a learning rate of 0.0005. We use batch gradient descent with a batch size of 64 for BERT and 16 for all other models. All models were trained until convergence with no early stopping.

4.3.2 Baseline and Feature Engineered Model

Our NN baseline model and NN NLP Features model (Approach 1) were implemented from scratch in Pytorch. Both were trained using a two layer neural network with 64 units in the first layer, 32 units in the second layer (with a ReLU layer added after each layer) and an output layer with 3 units (since our problem formulation has 3 classes). The spaCy library was used in order to extract tokens along with their syntatic dependency and part of speech information which were then used to compute NLP features.

4.3.3 Word Embedding Model

We experimented with 300-dimension GloVe and GN-GloVe embeddings [3]. With our basic neural net with hand-crafted features outlined, word embeddings were extracted for the two candidate mentions, the pronoun, the head word and the surrounding words for each. These embeddings were then reduced to 20-dimensions by averaging neighboring entries. Other dimensions were also experimented with. Word embedding extraction was written from scratch in Pytorch. See Figure 1 for model architecture.

4.3.4 Character Embedding Model

The character embedding model was impelmented from scratch in Pytorch though some code was adapted from Assignment 4/5 of the CS224N course. Character embeddngs were of size 8 and in the convolutional network a filter of size 4 was used. The output word embeddings were of size 32 and a dropout layer with $p = 0.1$ was used on the output of the highway network.

4.3.5 LSTM/RNN Model

Word embeddings for the entire text sequence plus padding tokens were fed into an LSTM. Hidden states for key words, including the candidate mentions, the pronoun, and their head words, were then extracted and used to compute a coreference score following the scoring mechanism outlined above from [2]. We experimented with hidden state dimensions, as well as adding in our NLP and distance features at different layers. We also experimented with reducing the sequence length in hopes of retaining more relevant contextual information, but this proved difficult as many examples contained the two candidates at the beginning and end of the sequence.

4.3.6 BERT/Transformer Model

The BERT/Transformer model used the pre-trained *bert-large-uncased* model from the *pytorch-pretrained-bert* package in python. This model represents each token as a 1024 dimensional embedding and we extracted the embeddings for three words - the pronoun and the two candidates. These extracted embeddings were then projected into a 64 dimensional hidden state with a dropout layer with probability 0.4. Other than the aforementioned python package, this was written from scratch in pytorch.

4.4 Results

See Table 3 for the improvement in masculine, feminine, bias, and overall F1 scores for all our approaches over the baseline models. In general, performance on F1 scores was correlated with a decrease in the log-loss metric as seen in Table 4 but there were a few cases were it was not such as in the Character Embedding model (Approach 3). The losses we obtained were in the same approximate range as losses found on the Kaggle leaderboard and some variance is expected as our final test set was different than the Kaggle submission test set.

The significant increase in performance over the baseline models from adding a set of hand crafted syntactic structure features (Approach 1) was slightly surprising to us as this score was already on par with some of the best off-the-shelf classifers. However, we do not have the task of clustering the mentions as the off-the-shelf resolvers do. Moreover, we trained and tested on the same GAP dataset while they trained on OntoNotes, highlighting the issue of overfitting to a particular dataset.

We see more incremental gains over the baseline when using pre-trained word embeddings which was expected as this approach had shown promising results in other co-reference models [5]. Character embeddings show slightly better improvement which also is reasonable as they are able to capture syntactic structures that are important to this type of task.

We did not see a noticeable increase in performance when using the Gender Neutral (GN) GloVe embeddings versus the regular GloVe embeddings. However, this is explained by the fact that the GN GloVe embeddings are created to reduce bias in cases of gender with respect to named entities (such as professions with stereotypical gender associations) but in the GAP dataset both possible candidates are names and always have the same gender so such information is less relevant. We also found that LSTMs performed worse than our simple model with pre-trained word embeddings which was also observed in [5] largely due to overfitting issues, long sequence lengths, and a relatively small dataset.

The best overall model was a combination of fine-tuning BERT along with hand crafted distance and syntactic structure features. This seems reasonable to us as BERT is one of the most powerful standalone language models today, has achieved success in other transfer learning settings and we had already seen strong improvements from our hand crafted features in Approach 1.

Table 3: Binary Classification F1 Scores by Gender

Model	M	F	B	O
Pretrained spaCy NeuralCoref Baseline	54.6	49.9	0.92	52.3
NN Baseline	57.2	58.3	1.02	57.7
NN NLP Features	69.1	67.9	0.98	68.5
NN NLP Features + GN-GloVe Word Embeddings	71.0	70.7	0.99	70.8
NN NLP Features + GloVe Word Embeddings	71.1	70.0	0.99	70.5
NN NLP Features + Character Embeddings	71.7	70.6	0.98	71.2
LSTM + NLP Features	69.2	70.6	1.02	69.9
BERT + NLP Features	74.0	71.7	0.97	72.8

Table 4: Overall Multiclass Logarithmic Loss

Model	Loss
Random Baseline	1.0986
NN Baseline	0.9423
NN NLP Features	0.7856
NN NLP Features + GN-GloVe Word Embeddings	0.7270
NN NLP Features + GloVe Word Embeddings	0.7686
NN NLP Features + Character Embeddings	0.7489
LSTM + NLP Features	0.7626
BERT + NLP Features	0.7124

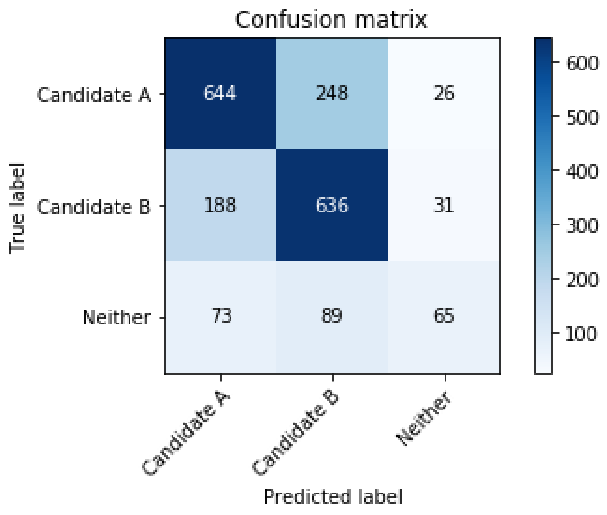

Figure 2: Confusion matrix with predictions from neural net model with GN-GloVe word embeddings and hand-crafted syntactic structure features

5 Analysis

5.1 Gender Bias

We were able to achieve nearly perfect bias scores of 1.0 ± 0.03 on all of our models. We attribute the lack of bias to training on a gender balanced dataset. Moreover, our spaCy baseline and BERT which are trained on gender imbalanced corpora had the lowest bias scores, further suggesting the importance of training on a gender balanced dataset for equal performance across genders.

5.2 Ablation Analysis

The examples with the best performance as determined by their loss are those where candidate A corefers with the pronoun, with candidate A beginning one sentence and the pronoun beginning the next sentence. An example of such follows with the pronoun in bold, each of the two candidate mentions italicized, and the candidate which corefers with the pronoun underlined.

> "*Philippe Burty* (6 February 1830 – 3 June 1890) was a French art critic. **He** contributed to the popularization of Japonism and the revival of etching, supported the Impressionsts, and published the letters of *Eug*ne Delacroix*."

The above sentence structure has very clear syntactic parallelism cues. These features are easily captured in our hand engineered NLP features as a mention's position within a sentence and that sentence's position within the document. To test this hypothesis, we removed sentence position features from our GN-GlOVe word embedding model and saw the loss on the test set increase from 0.7270 to 0.7555, showing a performance gain of 3.8% using the sentence position features.

5.3 Error Analysis

Our worst performance was on examples where neither candidate A nor candidate B corefers with the pronoun, many of which are difficult for even a human to understand, such as the example below.

> "Haufrecht embodied the premise, projecting a drowsy, fatigued lonesomeness with each action and word. The previous month, *Haufrecht* had garnered even stronger praise from Off-Off-Broadway Review's Doug DeVita as Common Basis staged another, less heralded premiere, *Grace Cavalieri's* Pinecrest Rest Haven: A frail-looking woman, **her** white hair tied up in a simple purple ribbon, enters a peach-and-white nursing-home waiting room and plaintively asks if anyone has seen her husband."

The low performance here can be explained by a lack of training examples for this class. Only approximately 10% of the train/test examples fall into the *neither* class (Figure 2). This examples highlights the syntactic complexity of the text in the GAP corpus and the challenge of this task.

6 Conclusion/Future Work

By using approaches inspired by various state-of-the-art coreference models in recent years, we have shown a significant improvement over a baseline model on the gendered balanced GAP dataset. Furthermore, the resulting F_1 scores outperform several proposed models referenced in the original GAP dataset paper [1]. We find that the best performing model combines both hand crafted distance and parallelism features along with fine-tuning a pre-trained BERT model and this outperforms other several approaches such as character and word embedding models.

One primary limitation of our work is that we did not experiment with training data sources other than the given GAP dataset. As such, one area for future work would be to use the full Wikipedia text instead of only the excerpt given in the dataset. This is potentially a very large data source and could allow us to train more complex models. Another idea for future work would be to train the end-to-end neural network model described in [2] on the larger OntoNotes dataset but try to *correct* the gender bias by replacing all pronouns with a gender neutral pronoun such as *ze*. We think that this may yield promising given that this model was the best off the shelf resolver on the GAP dataset as described in the original paper [1].

7 Additional Information

Our assigned project mentor is Xiaoxue Zang.

References

[1] Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A Balanced Corpus of Gendered Ambiguous Pronouns. arXiv preprint arXiv:1810.05201.

[2] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In Proceedings of EMNLP, pages 188–197, Copenhagen, Denmark.

[3] Zhao, J., Zhou, Y., Li, Z., Wang, W., and Chang, K. (2018b). Learning gender-neutral word embeddings. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018, pages 4847–4853.

[4] Honnibal, Matthew and Montani, Ines. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. 2017. Code: https://github.com/huggingface/neuralcoref

[5] Clark, K. and Manning, C. D. (2016). Improving coreference resolution by learning entity-level distributed representations. CoRR, abs/1606.01323.

[6] Hobbs, Jerry R., 1976. "Pronoun Resolution". Research Report 76-1, Department of Computer Sciences, City College, City University of New York. August 1976.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In Neural Information Processing Systems (NIPS), 2017

[8] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv e-prints.

[9] R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. arXiv:1505.00387, 2015.

[10] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2015. Character-Aware Neural Language Models. CoRR, abs/1508.06615.

[11] Hector J. Levesque. 2013. On our best behaviour. In Proc. AAAI.

[12] R. Jeffrey Pennington and C. Manning. Glove: Global vectors for word representation. 2014

[13] Weischedel, Ralph, et al. OntoNotes Release 5.0 LDC2013T19. Web Download. Philadelphia: Linguistic Data Consortium, 2013.