using Attentional RNNs for Local Named Entity Disambiguation

Anonymous ACL submission

Abstract

Hello, My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? My name is. what? my name is. who? Lorem ipsum.

1 Introduction

Named Entity Disambiguation (NED) is the task of disambiguating entity mentions within a fragment of text against a given knowledge base of entities and linking each mention to its intended entity. It has been recognized as an important downstream task for many NLP problems such as text categorization (Gabrilovich and Markovitch, 2007) and information retrieval (Dalton et al., 2014).

NED algorithms can broadly be divided into local and global approaches where local algorithms disambiguate mentions independently using local textual context while global approaches assume coherence among mentions within a single document and employ a coherency model to simultaneously disambiguate all mentions within a single document. Much attention was given recently to global algorithms (Ratinov et al., 2011a; Guo and Barbosa, 2014; Pershina et al., 2015) ¡- there's a ton of newer stuff which demonstrate very good performance on standard datasets. However while most standard datasets are based on text corpora such as news reports and Wikipedia paragraphs that are expected to be relatively long, well formed and coherent, some tasks such as fragments of web pages are generally shorter, noisier and less coherent. Such tasks require strong and noise-robust models.

Deep Neural Networks (DNN) have recently gained traction as state-of-the-art architectures that can model powerful data representations and complex feature interaction. DNNs achieve state-of-the-art results on a wide variety of tasks ranging from image classification (Krizhevsky et al., 2012) to machine translation (Bahdanau et al., 2014). He at el. (He et al., 2013a) used stacked auto-encoders to learn entity and context representations for NED. More recently Sun at el. and Francis-Landau at el. (Sun et al., 2015; **?**) have proposed models that incorporate convolutional neural networks for modeling and combining representations for a range of input signals and granularities.

We propose a novel Deep Learing based disambiguation algorithm where Recurrent Neural Networks (RNNs) with an attention mechanism are employed to model local context and discriminate correct entity assignments from corrupt ones. Recurrent Neural Networks are state-of-the-art language modeling architectures that are designed to model sequential data such as text. They have been demonstrated to give state-of-the-art results for NLP related tasks such as Neural Machine Translation (Bahdanau et al., 2014) and Language Modeling ? cite:??. Attention mechanisms are natural extensions for RNNs where the model can focus on the most relevant parts of the input text. They have been successfully employed for tasks such as Neural Machine Translation (Bahdanau et al., 2014) and Parsing (Vinyals et al.,

2015). We as well present a novel word2vec based method for initializing word and entity embeddings used by the model and demonstrate its importance. While training our model we further fine tune these embeddings via gradient descent and back-propagation.

We craft a large-scale dataset of short web fragments containing mentions disambiguated into Wikipedia by thousands of website editors. Our dataset is based on a subset of the Wikilinks dataset (Singh et al., 2012) which is re-purposed for NED. It represents large-scale and difficult task with count unique mentions disambiguated into $100K$ unique entities and where context is limited and noisy. We demonstrate our model greatly outperforms existing state-of-the-art NED algorithms on this dataset and in addition examine our algorithm on CoNLL (Hoffart et al., 2011), a standard NED dataset and show results comparable to other state-out-the-art methods on a smaller and cleaner dataset.

2 Methodology

We present our Attentional RNN model which is trained to a mentions context and a candidate entity, and estimate the likelihood of the candidate being correct rather then corrupt. We begin by describing the main Attentional RNN component. Then, we show how this model is trained for discriminating correct and corrupt entity assignments. Lastly we describe our method for initializing word and entity embedding.

2.1 Attentional RNN model

RNNs are state of the art.....

The challenge of modeling context is commonly addressed using word embeddings () add all refs from Malmud2014. Yet, in practice many of these studies refer the entire sentence as a simple collection or weighted average of its individual word embeddings, as they completely ignore its sequential structure. RNN are powerful models for learning the representation of wider contextual environments, which enables intelligent embedding of a sentence by gradually passing through its content and predicting some output using the system's history. In practice, it has been shown that copies of the context, which are opposite in direction, can be independently fed into a bidirectional RNN in order to improve prediction performance (Graves and Schmidhuber, 2004; Melamud and Goldberger, 2014). attention mechanisms are state of the art ....

We first focus on the mechanics of our Attentional RNN (ARNN) component as depicted in Figure 1. The Attentional RNN takes a window of words context as input and transforms it into a fixed length vector of features. It requires an embedded vector representation for a candidate entity to control its attention as well. This component is plugged into a larger DNN model and is trained by SGD and back-propagation along with the larger model. During the training process it learns to extract a useful feature representation of the context (given the candidate entity under consideration) for optimizing the loss function defined by the larger model. In our case predicting a ranking score for the candidate entity given the context. The basic component of our model is a standard RNN unit to process the context input. In our implementation we used standard GRU units (Cho et al., 2014) as these are state-of-the-art RNN models but any other RNN unit can be used as a drop-in replacement.

$$\begin{aligned} h_t &= f_{\Theta_1}(h_{t-1}, v_t) \\ o_t &= g_{\Theta_2}(h_t) \end{aligned} \tag{1}$$

Equation 1 represents the general semantics of an RNN unit. An RNN reads a sequence of vectors $\{v_t\}$ and maintains a hidden state vector $\{h_t\}$. At each step a new hidden state is computed from the previous hidden state and the next input vector by a function f parametrized by learnable parameters Θ_1. The output at each step is computed from the hidden state using a function g that is optionally parametrized by learnable parameters Θ_2.

A straight-forward use of an RNN in our model would be to train the RNN so that given a context as a sequence of word embedding vectors $\{v_t\}$, the last output o_t will be used to predict the ranking score of the candidate entity directly. However ??, ?? have shown attentional models are beneficial for NED which motivated us to incorporate a simple neural attention model. The attention model takes the entire sequence of intermediate output vectors of the RNN $\{o_t\}$ and computes a weighted average of the vectors governed by the equations shown in equation 2

$$
\begin{aligned}
& a_t \in \mathbb{R}; a_t = r_{\Theta_3}(o_t, c) \\
& a'_t = \frac{1}{\sum_{i=1}^{t} \exp\{a_t\}} \exp\{a_t\} \\
& o_{attn} = \sum_{i=1}^{t} a'_t o_t
\end{aligned} \tag{2}
$$

The main component in equation 2 is the function r_{Θ_3} which is parametrized by Θ_3 and computes an attention value for each word based on a controller c and the output of the RNN at step t. We then use a softmax to squeeze the attention values such that $\sum_{i=1}^{t} a'_i = 1$ and compute o_{attn}, the output of the attention model. The attention controller in our case is an embedding vector of the candidate entity itself. This allows the attention mechanism to decide how much individual words are important for ranking a given candidate. Our attention controller c is an embedding vector of the candidate entity and our attention function r is parametrized as a trainable single layer NN as shown in equation 3

$$
r_{\Theta_3}(o_t, c) = Ao_t + Bc \tag{3}
$$

2.2 Neural Network framework details and training regime AND MIXED NOTES

Our attentional RNN is incorporated in the NN framework as depicted in figure ???. We incorporate two separate RNNs, one for the left context and one for the right context. Note we feed the right context in reverse into RNN. Each RNN outputs a vector and these two vectors are concatenated along with the candidate entity embedding and are fed into a ranking DNN with a single hidden layer of size 300 and a relu non-linearity function, followed by an output layer with 2 neurons and a softmax non linearity. We train the neural network to optimize a cross entropy loss function where the first output neuron is set to predict the probability of the candidate entity being the correct one, and the second output neuron predicts the probability of the candidate entity not being the correct one. We have examined using a more conventional setting with a single output neuron trained using least squares loss, however this setting was found harder to train. We have implemented the entire model using the excellent Keras (Chollet, 2015) over Theano (Team et al., 2016) framework.

2.3 Training word and entity embeddings

Training our model implicitly trains its dictionaries of both word and entities by error back-propagation, however we have found that using pre-trained embeddings as a starting point improve model performance. To this end we have devised a Skip Gram with Negative Sampling (SGNS) (Mikolov et al., 2013) based training procedure that simultaneously trains both word and entity vectors in the same embedded space. We note Yamada at el. (Yamada et al., 2016a) have devised a somewhat similar and very successful method however they optimizes directly for disambiguation performance while we optimize for a simple, fast and plausible starting point for our deep model.

We use the word2vecf library by (Levy and Goldberg, 2014a) that is adapted from the original word2vec code[1] but allows to train directly on a dataset made of $(word, context)$ pairs rather then a textual corpus in a long string format. We exploit this and redefine the notion of $context$ from a context word to a context entity id. Specifically we take the corpus of Wikipedia pages as our knowledge base of entities and create a dataset where each page identified by the internal $pageid$ in Wikipedia database is scanned and split into a list of words $\{word_i\}$. We then add the pair $(word_i, pageid)$ for each word in each page in Wikipedia.

As shown by Levy and Goldberg (Levy and Goldberg, 2014b) training embeddings on this dataset using SGNS produces word and entity embedding that implicitly factorize the PPMI matrix of word-entity co-occurrence matrix. This well-known matrix is closely related to the TFIDF word-entity matrix used by (Gabrilovich and Markovitch, 2007) in Explicit Semantic Analysis and was found useful in a wide array of NLP tasks.

Prior to training we filtered words and documents (entities) appearing less then 20 in the dataset. We trained embeddings of length 300 for 10 iterations over the dataset. We used default values for all other parameters.

what was the limit i used for minimum length of articles/minimum freq of words?

words are close to entities they are indicative of. Words appearing in similar entity pages are grouped together. Entities with similar pages are grouped together

[1] Available at https://code.google.com/archive/p/word2vec/

show results of the analogies experiment we did indicating semantic structure for the WORD vectors

3 Web-Fragment based NED Dataset

We introduce a new large scale NED dataset for web-fragment crawled from the web. Our dataset is derived from the Wikilinks dataset originally collected at Google by Singh at el. (Singh et al., 2012) for a cross-document co-reference task. cross-document co-reference entails clustering mentions referring to the same entity across a set of documents without consulting a predefined knowledge base of entities, and is many-a-time regarded as a downstream task for knowledge base population (KBP). Wikilinks is constructed from crawling websites and collecting hyperlinks into Wikipedia and the web context they appear in. These hyperlinks are assumed to be ground-truths for linking mentions (hyperlink anchor-text) to distinct entities (Wikipedia pages). Wikilinks contains 40 million mentions covering 3 million entities collected from over 10 million web pages.

Despite the difference between a NED and a co-reference task, we believe the nature of Wikilinks makes it directly applicable to NED as well. It can be seen as a large-scale, naturally-occurring and crowd-sourced dataset where thousands of human annotators provide ground truth for mentions of interest. Its web sourcing entails an added interest since it contains every kind of noise expected from automatically gathered web content, including many faulty, misleading and peculiar ground truth labels on the one hand, and on the other hand noisy, malformed and incoherent textual context for mentions.

While noise in crowd-sourced data is arguably a necessary trade-off for quantity, we believe the contextual noise represents an interesting test case supplementing existing standard datasets for NED such as CoNLL (Hoffart et al., 2011) and ACE and Wiki (Ratinov et al., 2011a) as these are sourced from mostly coherent and well formed text such as news articles and Wikipedia pages. Such a test case emphasize utilizing strong and adaptive local disambiguation techniques, and marginalizes the utility of coherency based global approaches.

elaborate on intralink dataset

3.1 Preprocessing the Wikilinks dataset

In order to utilize the Wikilinks dataset for NED we have first performed a number of filtering and preprocessing steps. Since we have opted for focusing on local disambiguation we choose a version of Wikilinks with only local contexts for mentions and URLs to the containing page rather then the much larger version with full pages [2]. This sums to around 5Gb of compressed data (compared to 180Gb for the full texts).

The first preprocessing step we took was resolving ground-truth links using a $7/4/2016$ dump of the Wikipedia database [3]. The same dump was consistently used throughout this research. We used the $page$ and $redirect$ tables for resolution and kept the database id as a unique identifier for Wikipedia pages (entities). We discarded mentions where the ground-truth could not be resolved. This resulted in retaining over 97% of the mentions. We then permuted the order of mentions within the data and split it into train, evaluation and test set. We split the data $90\%/10\%/10\%$ respectively. Since websites might include duplicate or closely related content we did not assign mentions into splits on an individual base but rather collected all origin domains and assigned each domain along with all mentions collected from it into a split collectively.

The last and most important task was filtering the data. We first counted how many times each mention refers to each entity: $\#\{e|m\}$ and calculated the prior conditional-probability of an entity given a mention: $p(e|m) = \#\{e|m\}/\sum_{e'} \#\{e'|m\}$. Examining these distributions revealed most mentions either had very little ambiguity or appeared very few times and had a number of ground truths each appearing just a handful of times. We have deemed unambiguous mentions to be not-interesting and scarce mentions to be suspected as noise with a high probability and designed a procedure to filter both this cases. We therefore decided to retain only mentions for whom at least two ground-truth entities have $\#\{e|m\} \geq 10$ and $p(e|m) \geq 0.1$.

This procedure aggressively filtered the dataset and we were left with $2.6M$ training, $300K$ test and $300K$ evaluation samples. We believe that

[2] Both available at http://www.iesl.cs.umass.edu/data/wiki-links

[3] Recent Wikipedia dumps are found at https://dumps.wikimedia.org/

doing so filters uninteresting cases while emitting a dataset that is large scale yet manageable in size for research purposes. We note that we have considered filtering $(mention, entity)$ pairs where $\#\{entity|mention\} \leq 3$ since these are suspected as additional noise however decided against this procedure as it might filter out long tail entities, a case which was deemed interesting by the community.

4 Experiments

4.1 Training details

We now describe our setup for evaluating the proposed model for the task of NED. Following the motivation for disambiguating the Wikilinks data set, which was explained in section ReferencesWeb-Fragment based NED Dataset, we trained our model on $2.6M$ mentions. This procedure was carried out with a 20-core CPU machine during XX days add description of the machine and timing.

Prior to the training we generated a basic counts/statistics file of mentions and entities from our corpus in a statistics file, for the creation of additional features and filtering mentions. [What else?]. Specifically, it held a mention-count, an entity-count, a mention-links with the distribution over the possible candidates and a count for all words appearing in the context of a mention.

Due to framework limitations we have trained our RNNs with a fixed size left and right contexts. These where set a 20 word window to each side of the mention. When the context is shorter on either side, we pad it with a special PAD symbol. We use a stop-word list to filter less informative stopwords.

The input in the model was streamed as pairs of entity-candidate. Following Mikolov at el. (Mikolov et al., 2013) we used a Negative Sampling approach where for each positive example the network is trained on, it is also trained on k negative samples. In this framework, the context remains the same , while incorrect candidates are randomly sampled from all possible entities. We deviate from Mikolov , which picks negative examples according to their distribution in the training corpus, by uniform sampling from all possible entities . double check - In uniform sampling the ratio of positive to negative samples for each entity is proportional to the prior probability of the entity within the corpus, which bias the network toward ranking common entities higher.

The optimization of the model was carried out using standard back propagation and AdaGrad optimizer(Duchi et al., 2011). We allowed the error to propagate through all parts of the network and fine tune all trainable parameters, including fine-tuning the word and entity embeddings themselves. The scale of the data enabled us to run only 1 epoch over the data.

CoNLL 2003 NER data is a an evaluation that is commonly used in research for benchmarking NED solutions (Globerson et al., 2016; Hachey et al., 2013; Yamada et al., 2016b; Pershina, 2015). This has driven us to hold additional models, trained on 18505 non-NIL mentions from the CoNLLs training set, and evaluated on test-b set with 4485 cases.

CoNLL The CoNLL corpus is a public, free and annotated dataset, which is compromised out of a relatively large amount of news articles written in 1996 by the Reuters newswire. It shares 1393 documents from two time periods of overall 12 days, which are split into train, development and test set according to $68 - 15 - 17\%$. The performance on the corpus is usually measured using micro accuracy (p@1), which is a the average over the label of the mentions, having the golden-sense in their candidate set. While disregarding the performance of the candidate generation process, this measure can be delusive when the recall of the candidate generation is low. Therefore, it is common to use candidate generation resources, such as YAGO (Hoffart et al., 2011) or PPRforNED (Pershina, 2015), with over 99% candidate recall for better comparability of the results. Another broadly used measure is the macro accuracy, which averages the precision of all documents.Typical and recent p@1 results of local disambiguation approaches on the test-b dataset have ranged between 80% to 89% (see table **??**), when very recently the 90% barrier was broken by Yamada et al.(Yamada et al., 2016b). Results on CoNLL appear sometimes in literature under the AIDA system, which is an online disambiguation system offered by Hoffart that contains the CoNLL 2003 NER task.

4.2 Learning to rank

We had two kinds of models trained on the CoNLL corpus. Initially, we used the pure Attentional RNN architecture, trained over 20% is this right? of the intralinks corpus with fine-tuning. Because

of the scale of the data we used only one epoch. Inspired by Yamada's approach, we extended our model to be part of a parent Gradient Boosting Regression Tree (GBRT) classifier that is known for demonstrating neat performance in ranking applications (**?**). For every data fragment, we took the preceding model's prediction, which is the probability that a candidate suits the input mention given the context, and fed it into a feature vector. In this manner we generated an input vector for every pair of mention-candidate from the CoNLL training set and let the model iterate over the entire data. Practically, we used sklearn's GradientBoostingClassifier implementation (**?**) with parameters similar to those reported by Yamada. We trained with the logistic-regression deviance and set the learning rate, number of estimator and maximum depth of a tree to 0.02, 10000 and 4, respectively. For the CoNLL experiments we also added Yamada's base and string similarity features into the input vector. We find it important to note that in contrast to many other studies, we didn't focus on surpassing current state of the art results on the CoNLL task, as it is a precisely edited and specific newspaper corpus with stories covering merely two weeks of reports (Sang and De Meulder, 2003).

4.3 Count and string based features necessary??

Besides of embeddings of the context and entities' the model's input consisted several more features. We implemented the base features that were introduced by Yamada using statistics from Wikipedia that were computed prior to the training. In this manner we obtained the conditional and marginal entity prior, $P(e|m)$ and $P(e)$, which are normalized measures giving the number of times entity e was linked to a specific or to any mention m in the dataset. Information about the source document was captured by the max prior feature that is takes the maximum prior probability of

4.4 Comparison with other methods

We use our the suggested Wikilinks corpus from section ReferencesWeb-Fragment based NED Dataset to compare the results of our model with two state of the art and well established algorithms.

- **Yamada et al.** (Yamada et al., 2016b) have published a recent paper that introduces the highest score of precision@1 on CoNLL test-b. For this reason we used this work as a principle guideline for the evaluation over the CoNLL corpus. The paper describes an embedding method that learns the relatedness of entities and jointly maps entities and context to the same vector space using the skip-gram model. The distance between these embeddings is then computed to form several features of an input vector used for point-wise training an GBRT model. Full Yamada refers to the complete algorithm, including coherence and the two-step approach, while Yamada (no coherence) is the partial version, which excludes the former steps.

- **Cheng et al.** (Cheng and Roth, 2013) addressed the Disambiguation to Wikipedia, or the "Wikificaiton" task, with a combination of local and global solutions. In this model, mentions are disambiguated locally by generating for each of them a ranked list of candidates using the GLOW Wikification system offered by Ratinov et al. (Ratinov et al., 2011b). We compare our results to the ranking step of the algorithm, were a linear Ranking Support Vector Machine is trained over a set of local and global features to return the list of candidate sorted by their likelihood. We compared our results to the Ranker accuracy metric , which is relevant an evaluation metric equivalent to the p@1 score.

As a baseline we took the standard Most Probable Sense (MPS) prediction, which corresponds to the $\arg\max_{e \in E} P(e|m)$, where E is the group of all possible entities. In addition to the former models we also present performance on CoNLL of the following papers - Lazic et al. (Lazic et al., 2015), Francis-Landau et al. (Francis-Landau et al., 2016), He et al. (He et al., 2013b), Hoffart et al. (Hoffart et al., 2011) and Chisholm et al. (Chisholm and Hachey, 2015) ,as they are strong local approaches which demonstrate relative good results.

4.5 Results

The micro and macro accuracy scores on CoNLL test-b are displayed in table 1. We used the *PPRforNED* dataset (Pershina, 2015) for extracting the candidates for each mention, as it yielded 99.7% candidate-generation recall and provided better results compared to YAGO.

CoNLL test-b		
Model	Micro accuracy	Macro accuracy
ARNN	**84.2**	**85**
GBRT: Base + ARNN features	**85.6**	**88**
Yamada et al. (no coherence)	90.9	92.4
Lazic et al.	86.4	-
Francis-Landau et al.	85.5	-
He et al.	84.82	83.37
Hoffart et al.	79.57	80.71
Chisholm et al.	88.7	-
Baseline (MPS)	77	77

Table 1: Evaluation on CoNLL. Bold font denotes the models offered in this study

In the following table we see performance of models on the Wikilinks test set. With this corpus we report on 97.3% candidate generation recall, having in average 13.5 candidates per mention.

Wikilinks test set	
Model	**Micro accuracy**
ARNN	84.2
GBRT: Base + ARNN features	66.8
Yamada (no coherence)	-
Update results	...
Baseline (MPS)	55.9

Table 2: Evaluation on Web-Fragment data (Wikilinks)

** add insights regarding the results and comparison

4.6 Methods sensitivity/study

elaborate on:
** ESA embedding initialization
** intra vs. wikilinks training
** cost function (max-margin vs. binary crossentropy)
** downsampling

5 Conclusions

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Xiao Cheng and Dan Roth. 2013. Relational Inference for Wikification. *Empirical Methods in Natural Language Processing*, (October):1787–1796.

Andrew Chisholm and Ben Hachey. 2015. Entity Disambiguation with Web Links. *Transactions of the Association for Computational Linguistics*, 3(0):145–156.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Franois Chollet. 2015. keras. `https://github.com/fchollet/keras`.

Jeffrey Dalton, Laura Dietz, and James Allan. 2014. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 365–374. ACM.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Matthew Francis-Landau, Greg Durrett, and Dan Klein. 2016. Capturing Semantic Similarity for Entity Linking with Convolutional Neural Networks. pages 1256–1261.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJcAI*, volume 7, pages 1606–1611.

Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2016. Collective Entity Resolution with Multi-Focal Attention. *Acl*, pages 621–631.

Alex Graves and Jurgen Schmidhuber. 2004. Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures.

Zhaochen Guo and Denilson Barbosa. 2014. Entity linking with a unified semantic representation. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 1305–1310. ACM.

Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R. Curran. 2013. Evaluating entity linking with wikipedia. *Artificial Intelligence*, 194:130–150.

Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. 2013a. Learning entity representation for entity disambiguation. In *ACL (2)*, pages 30–34.

Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. 2013b. Learning Entity Representation for Entity Disambiguation. pages 30–34.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Nevena Lazic, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2015. Plato: A Selective Context Model for Entity Resolution. *Transactions of the Association for Computational Linguistics*, 3:503–515.

Omer Levy and Yoav Goldberg. 2014a. Dependency-based word embeddings. In *ACL (2)*, pages 302–308.

Omer Levy and Yoav Goldberg. 2014b. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185.

Oren Melamud and Jacob Goldberger. 2014. Learning Generic Context Embedding with Bidirectional LSTM.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Maria Pershina, Yifan He, and Ralph Grishman. 2015. Personalized page rank for named entity disambiguation. In *Proc. 2015 Annual Conference of the North American Chapter of the ACL, NAACL HLT*, volume 14, pages 238–243.

Maria Pershina. 2015. Personalized Page Rank for Named Entity Disambiguation. (Section 4):238–243.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011a. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384. Association for Computational Linguistics.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011b. Local and Global Algorithms for Disambiguation to Wikipedia. *Acl 2011*, 1:1375–1384.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *CONLL '03 Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, 4:142–147.

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2012. Wikilinks: A large-scale cross-document coreference corpus labeled via links to Wikipedia. Technical Report UM-CS-2012-015.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling mention, context and entity with neural networks for entity disambiguation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1333–1339.

The Theano Development Team, Rami Al-Rfou, Guillaume Alain, Amjad Almahairi, Christof Angermueller, Dzmitry Bahdanau, Nicolas Ballas, Frédéric Bastien, Justin Bayer, Anatoly Belikov, et al. 2016. Theano: A python framework for fast computation of mathematical expressions. *arXiv preprint arXiv:1605.02688*.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, pages 2773–2781.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016a. Joint learning of the embedding of words and entities for named entity disambiguation. *arXiv preprint arXiv:1601.01343*.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016b. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. *arXiv preprint arXiv:1601.01343*, page 10.