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 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 some coherence among mentions within a single document and therefore employ a coherency model to simultaneously disambiguate many mentions. Much attention was given recently to global algorithms (Ratinov et al., 2011a; Guo and Barbosa, 2014; Pershina et al., 2015) (there's 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, other corpora such as fragments of web pages can be shorter, noisier and less coherent.

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; Francis-Landau et al., 2016a) 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-Learning 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. differentiate RNN from CNN. Why are their better? Recurrent Neural Networks are state-of-the-art language modeling architectures that are designed to model sequential data such as text and 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 is trained to focus on the most relevant parts of the input text. Attention mechanisms have been successfully employed for tasks such as Neural Machine Translation (Bahdanau et al., 2014) and Parsing (Vinyals et al., 2015).

We present a novel word2vec (Mikolov et al.,

2013) based method for initializing word and entity embeddings employed 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 a 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

The architecture for our model is presented in Figure 1. The model takes a window-of-words to the left and right of a mention as its textual context and some candidate entity. The left and right contexts are fed into a duo of Attentional RNNs which process each side and produce a fixed length vector representation. Notice the right side is fed backwards into the model. Each Attentional RNN also uses the candidate entity to control its attention which allows it to attend to the most discriminating parts of the context given the candidate at hand.

The output vectors generated by both Attentional RNNs and the embedding of the entity itself are then fed into a classifier network consisting of a hidden layer, and an output layer with two output units and softmax activation. The output units are trained to emit the likelihood of the candidate being a correct or corrupt assignment. We optimize our model end-to-end with a cross-entropy loss function.

We have set the hidden layer size to be 300 and used a ReLU activation for this layer. We have found the width and depth of the classifier to be of little impact on performance, but using a ReLU non-linearity was found to be important. We have also applied dropout with $p = 0.5$ to the hidden layer.

2.1 Attentional RNN component

The mechanics of our Attentional RNN component are depicted in Figure 2. In our implementation we used a standard GRU unit (Cho et al., 2014) as it is a state-of-the-art RNN model but any other RNN unit can be used as a drop-in replacement.

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 Θ_1. The output at each step is computed from the hidden state using a function g that is sometimes parametrized by Θ_2. This allows the RNN to 'remember' important signals while scanning the context and to recognize signals spanning multiple words.

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

While an RNN unit can be used as-is in our model by feeding the last output vector o_t directly into the classifier network, we have implemented an attention mechanism that allows the model to be aware of the candidate entity it is evaluating when computing an output vector. Equation 2 details the equations governing the attention model.

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

The main component in equation 2 is the function r, parametrized by Θ_3, which computes an attention value at each step using $v_{candidate}$, the candidate entity embedding, as a control signal. We then use a softmax to squeeze the attention values such that $\sum_{i=1}^{t} a'_i = 1$ and compute the final output o_{attn} as a weighted sum of all the output vectors of the RNN. This allows the attention mechanism to decide on the importance of different context parts when examining a specific candidate.Noam: the following equation is not clear. explain A,B and b. connect to the storyline...

$$r_{\Theta_3}(o_t, v_{candidate}) = Ao_t + Bv_{candidate} + b \tag{3}$$

We parametrize our attention function r as a single layer NN as shown in equation 3

2.2 Training initial word and entity embeddings

Training our model implicitly trains its dictionaries of both word and entity embedding by error back-propagation. However, as will be shown in section 4, we have found using pre-trained embeddings to significantly 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 method however our method trains much faster and requires only the textual content of Wikipedia pages while they require Wikipedia cross-references and category structure as well.

We use the word2vecf library[1] by Levy and Goldberg (Levy and Goldberg, 2014a) that is adapted from word2vec code and allows to train on a dataset made of $(word, context)$ pairs rather then a textual corpus in string format, as is done in the original word2vec. We exploit this to redefine $context$ as a context entity rather then a contextual word.

We do this by considering each page in Wikipedia to represent a unique entity, enumerated by the $pageid$ identifier in Wikipedia database and having a textual description (the page itself). For each word $\{word_i\}$ in the page we add the pair $(word_i, pageid)$ to our dataset. We however limit our vocabularies by ignoring both rare words that appear less then 20 times and entities that have less then 20 words in their description.

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 word-entity co-occurrence PPMI matrix. This matrix is closely related to the TFIDF word-entity matrix used by Gabrilovich and Markovitch (Gabrilovich and Markovitch, 2007) in Explicit Semantic Analysis and found to be useful in a wide array of NLP tasks.

For our experiments we trained embeddings of length 300 for 10 iterations over the dataset. We used default values for all other parameters in word2vec.

-needs developing-

-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 of web-fragments crawled from the web. Our dataset is derived from the Wikilinks dataset originally collected 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 was constructed by crawling the web and collecting hyperlinks linking to Wikipedia and the web context they appear in. The anchor texts act as mentions and the link targets in Wikipedia act as ground-truths. Wikilinks contains 40 million mentions covering 3 million entities and 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 valuable to NED as well. This was recognized by Chisholm and Hachey (Chisholm and Hachey, 2015a) who examined collecting statistics from Wikilinks for a NED system. However we are interested in Wikilinks as a test case.

Wikilinks can be seen as a large-scale, naturally-occurring and crowd-sourced dataset where thousands of human annotators provide ground-truths for mentions of interest. Its web sourcing entails 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 in particular represents an interesting test-case that supplements existing standard datasets such as CoNLL (Hoffart et al., 2011), ACE and Wiki (Ratinov et al., 2011a) as these are all sourced from mostly coherent and well formed text such as news articles and Wikipedia pages. Wikilinks emphasizes utilizing strong and adaptive local disambiguation techniques, and marginalizes the utility of coherency

[1] Available at https://bitbucket.org/yoavgo/word2vecf

based global approaches.

3.1 Preprocessing the Wikilinks dataset

We detail a number of filtering and processing steps used to adapt Wikilinks as a test dataset for NED. The original dataset exists in a number of formats, and we have chosen a version with only short local contexts[2] since it renders the size of the dataset a manageable 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 *pageid* column as a unique identifier for Wikipedia pages (entities). We discarded mentions where the ground-truth could not be resolved, resulting in retention of over 97% of the mentions. We then randomly 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 basis 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 collected all pairs of mention m and entity e appearing in the dataset and computed the following two statistics: how many times m refers to e: $\#\{e|m\}$ and the conditional probability of e given m: $p(e|m) = \#\{e|m\}/\sum_{e'} \#\{e'|m\}$. Examining these distributions revealed many mentions belong to two extremes: either they had very little ambiguity or had a number of candidate entities each appearing very few times. We have deemed the former to be unambiguous and not-interesting, and the latter to be suspected as noise with high probability. We therefore designed a procedure to filter both this cases: We retained 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 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 (m, e) pairs where $\#\{e|m\} \leq 3$ 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

In this section the performance of our model is being evaluated for the task of NED. We detail the training techniques of different ARNN based models for several disambiguation corpora. All models were implemented in Python using the excellent Keras (Chollet, 2015) over Theano (Team et al., 2016) and Tensorflow (Abadi et al., 2015) frameworks.

4.1.1 Training on Wikilinks

Following the motivations that were described in section 3, we wanted to test the efficiency of our ARNN solution when trained over the Wikilinks corpus. We started by generating a basic file with statistics of mentions and entities that appeared in our corpus. This file was used later for creating additional features and filtering irrelevant mentions Noam: is it used for something else?. Specifically, it held the mentions and entities count; a mention-link structure with a histogram of candidates; and the number of times each context word appeared in the corpus.

We have trained our GRUs with fixed size left and right contexts, using a 20 word window to each side of the mention. In cases were the context was shorter than the fixed size, we padded it with a special PAD symbol. Further, we filtered out stop words according to NLTK's stop-word list.

The input to the model was streamed as pairs of entity and candidate. The issue of having only true-matching pairs of mentions and entities in our corpus, was addressed by extending our training set to include also false-matching, or corrupted, examples. This was accomplished by randomly sampling k entities for each mention ,hence obtaining k corrupted training pairs. This framework is somewhat similar to the negative sampling approach introduced by Mikilov et al. (Mikolov et al., 2013), however instead of sampling the false examples according to the entity distribution, we

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

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

used uniform sampling. With uniform sampling the ratio of true-to-corrupted samples becomes higher for entities with strong prior. This property biases the network toward preferring entities that are more frequent in the corpus.

The optimization of the model was carried out using standard back propagation and an AdaGrad optimizer (Duchi et al., 2011). We allowed the error to propagate through all parts of the network and to fine tune all trainable parameters, including the word and entity embeddings themselves. A single epoch with $2.6M$ mentions was used for training the model during 2 Noam: edit runtime days using a 20-core CPU machine.

4.1.2 Training on CoNLL

CoNLL 2003 NER corpus 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 motivated us to hold additional ARNN based models for comparing our method with state of the art results. The CoNLL corpus is a public, free and annotated dataset, which is compromised out of a relatively large amount of news articles written by the Reuters newswire in 1996. It shares 1393 documents from overall 12 days of news reports that are split into train, development and test sets according to a ratio of 68%-15%-17%.

The performance on CoNLL is usually measured using micro accuracy (p@1), which denotes the accuracy of the task, over all the mentions that have their golden-sense in their candidate set. In practice, p@1 disregards the performance of the candidate generation process from the results. Thus, it may be seen as a delusive metric when the recall of the candidate generation is low. Therefore, it is common to use resources, such as YAGO (Hoffart et al., 2011) or PPRforNED (Pershina, 2015), which can be exploited as candidate list generators with over 99% candidate generation recall. Macro accuracy is another broadly used evaluation measure, which averages the precision over all the documents of the corpus. In some papers, the performance on CoNLL appears as results on AIDA, which is an online disambiguation system offered by Hoffart et al. (Hoffart et al., 2011) that contains the CoNLL 2003 NER task.

CoNLL's training set has in total 18505 non-NIL mentions, which is an insufficient amount of data for training our suggested ARNN model given that it has over 700K parameters. For this reason we have produced an additional intralinks corpus, which is fabricated out of Wikipedia's conferences and covers the entire entity space of CoNLL. We sampled 20% of this enormous corpus for initial training of the ARNN model. Then, we fine-tuned the network using 1 epoch of the CoNLL's training set.

Inspired by Yamada's approach, we have also experimented with injecting the predictions of the preceding model into a well established classification model. Specifically, we employed the Gradient Boosting Regression Tree (GBRT) classifier, a state of the art ensemble model that is based upon boosting results by stacking weak regression trees together (Friedman, 2001). For every data fragment we took the ARNN's output probability, that a candidate matches the input mention given its context, and added it into a feature vector. This vector also featured the base and string similarity features that are detailed in Yamada et al. (Yamada et al., 2016a). We used sklearn's GradientBoostingClassifier implementation (Pedregosa et al., 2011) 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. The models were evaluated with CoNLL's test-b set which shares 4485 non-NIL disambiguation cases.

4.2 Comparison with other methods

We emphasize that contrary to 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). Instead we were more interested in comparing strong NED models over a more naturally-occurring corpus such as Wikilinks. Therefore, we compare the former models over both CoNLL and Wikilinks 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 set. For this reason we used Yamada's study as a guideline for training and evaluating over CoNLL. The paper describes an embedding method that learns relatedness between entities and jointly maps entities and context to the same vector space using the skip-gram model. The distance between those embeddings is then computed to generate features, which are utilized for point-wise training a GBRT model. The full algorithm consists various levels and features, including features for coherence and a two-step global training scheme. As this work focuses on local disambiguation, we compare it to a partial version of Yamada's algorithms, 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 a ranked list of candidates for each mention 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 candidates sorted by their likelihood. Specifically, we examine the Ranker accuracy metric , which is 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. We also display the evaluation on CoNLL test-b for the following papers - Lazic et al. (Lazic et al., 2015), Francis-Landau et al. (Francis-Landau et al., 2016b), He et al. (He et al., 2013b), Hoffart et al. (Hoffart et al., 2011) and Chisholm et al. (Chisholm and Hachey, 2015b) ,as they are all strong local approaches and a good source for comparison.

4.3 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 of each mention, as it yielded 99.7% candidate-generation recall and provided better results compared to YAGO.

Table 2 includes the evaluation on the Wikilinks test set. In this corpus we report on 97.3% candidate generation recall, having in average 13.5 candidates per mention.

* needs to be added
** needs to be updated

CoNLL test-b		
Model	Micro accuracy	Macro accuracy
ARNN	**84.2**	**85**
GBRT: Base + ARNN features	**85.6**	**88**
Yamada et al. (partial)	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

Wikilinks test set	
Model	**Micro accuracy**
ARNN	**64.8**
GBRT: Base + ARNN features	**66.8**
Yamada et al.(partial)	59.8 **
Cheng et al.	*
Baseline (MPS)	55.9

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

add insights regarding the results and comparison

4.4 Model sensitivity

Noam: this should be extended elaborate on:
!! ESA embedding initialization
!! attention vs. no attention

Wikilinks test set	
Model	**Micro accuracy**
ARNN w/o ESA init.	61
ARNN w/ ESA init. w/o Attention	64.1
ARNN w/ ESA & Attention	64.8

Table 3: ARNN Model sensetivity

5 Conclusions

References

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

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. 2015a. Entity disambiguation with web links. *Transactions of the Association for Computational Linguistics*, 3:145–156.

Andrew Chisholm and Ben Hachey. 2015b. 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. 2016a. Capturing semantic similarity for entity linking with convolutional neural networks. *arXiv preprint arXiv:1604.00734*.

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

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

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.

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.

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.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830.

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.