Biomedical Named Entity Recognition Using Neural Networks

George Mavromatis
Stanford University
gmavrom@gmail.com

Abstract

We investigate the task of Named Entity Recognition (NER) in the domain of biomedical text. There is little published work employing modern neural network techniques in this domain, probably due to the small sizes of human-labeled data sets, as non-trivial neural models would have great difficulty avoiding overfitting. In this work we follow a semi-supervised learning approach: We first train state-of-the art (deep) neural networks on a large corpus of noisy machine-labeled data, then "transfer" and fine-tune the learned model on two higher-quality human-labeled data sets. This approach yields higher performance than the current best published systems for the class DISEASE. It trails but is not far from the currently best systems for the class CHEM.

1 Introduction

The NER task in biomedical text differs from general NER in several ways. The large number of synonyms and alternate spellings of an entity cause explosion of word vocabulary sizes and reduce the efficiency of gazetteers. Entities often consist of long sequences of tokens, making harder to detect boundaries exactly. It is very common to refer to entities also by abbreviations, sometimes non-standard and defined inside the text. Polysemy or ambiguity is pronounced: proteins (normally class GENE) are also chemical components and depending on the context occasionally should be classified as class CHEM; tokens that are sometimes of class SPECIES can be part of a longer entity of class DISEASE referring to the disease caused by the organism or the speciliazaion of disease on the patient species.

It is also important to distinguish between the Named Entity Detection and the Named Entity Disambiguation of Normalization tasks. The former refers to determining in the text the start and ending positions of entities, the latter mapping them to a taxonomy of concepts within the entity class. In the Disambiguation task the exact positions in the text may or may not need to reported, so it is not necessarily a harder superset of NER. This project is only concerned with the NER task.

There is practical commercial need for automation on mining the massive literature for biomedical research or practice. The industry term "scientific curator" refers to a person going through the literature and whose job can be greatly assisted with a NER system.

2 Background

The current state-of-the-art NER systems are feature-rich supervised learning classifiers, with significant domain-specific feature engineering. Most competitive systems use Conditional Random Fields [1] classifiers. Features are usually character-level N-grams, part-of-speech tags, bag-of-words features and membership in gazeteers. The first papers using some elements of modern neural networks were published in 2015 ([10]). The small size of human-annotated corpora makes it very hard to train non-trivial neural models.

3 Description of Named Entities

We are interested in the following 4 classes:

- GENE (genes and proteins are equivalent): Examples: symbols "G6PD", "Bcl-2", long names: "glucose-6-phosphate dehydrogenase", "epidermal growth factor receptor", "interleukin-5"
- DISEASE: Examples: "acromegaly", "rheumatoid arthritis" or "RA", "adenocarcinoma of the lung", "Li-Fraumeni syndrome", "adult onset globoid cell leukodystrophy" or "GCL"
- CHEM for "Chemical": Examples: "triglycerides", "Naproxen", "AMN082", "7-beta-hydroxy-harpagide", "NaOH", "Pb", "iron", "2,3,7,8-Tetrachlorodibenzo-p-dioxin"
- SPECIES: Examples: "human", "mice", "Neospora caninum", "N. caninum"

The following example is an extract from Pubmed ID 26522111. Entities in italics.

"TITLE: Detection of *human papillomavirus* in *squamous cell carcinoma* arising from *dermoid cysts*. OBJECTIVE: *Primary squamous cell carcinoma (SCC) of the ovary* in *humans* is rare ... *Human papillomavirus (HPV) infection* is a critical factor that induces *tumor* formation..."

The tokens "human papillomavirus" appear both as a standalone SPECIES entity and inside the "Human papillomavirus (HPV) infection" DISEASE entity. Token "Humans" appears as a standalone SPECIES entity and inside two other entities. Although "Primary squamous cell carcinoma (SCC) of the ovary" is the most proper DISEASE entity, "squamous cell carcinoma" and "SCC" are very plausible, but plain "carcinoma" is not. These highlight the complexities with multi-token entities and attached or nested words or entities.

4 Datasets

4.1 Machine-Annotated Data Set

Pubmed [2] is a collection of about 20 million unlabeled biomedical abstracts. A team at National Center for Biotechnology Information (NCBI) attaches entity labels using state-of-art systems for labels GENE (GNormPlus extension of [3]), DISEASE [4], CHEM [5], and SPECIES: SR4GN. By inspection of more than 100 abstracts,it is clear that there are significant errors in the annotation process. For instance some abstracts are missing labels that the component systems are known to produce and they do in most other abstracts in the same corpus. The class CHEM is especially poorly labeled with very low recall for multi-token entities. Still, the data set is of acceptable quality for the semi-supervised learning task of this work.

4.2 Human-Annotated Data sets

- NCBI disease corpus [7]: 593 Training, 100 development, 100 test abstracts. Labels: DISEASE. Not very general, very small, high quality otherwise.
- BioCreative IV CHEMDNER corpus [8]: 3500 training, 3500 development, 3000 test abstracts. Labels: CHEM. Very high quality, adequate size.
- BioCreative V Chemical Disease Relation (CDR) corpus [9]: 950 training and development, 500 test abstracts. Labels: DISEASE and CHEM. Designed for an entity-relationship task.

We have confirmed that the benchmark systems applied on the data set in 4.1 have *not* been trained on the test set of above corpora.

5 Methodology and Results

5.1 Tokenization and Vocabularies

For the word level representation, abstracts were tokenized with the Stanford Core NLP tokenizer [11] and were further split on hyphenation (more appropriate for biomedical NER). Tokens not all in uppercase were lowercased, but all-uppercase tokens were left as distinct words. All numbers were changed to 0. The goal of this preprocessing is to drop information of little use to the NER classifier and at the same time reduce the word vocabulary size dramatically. Similar preprocessing steps were used in [3] and [16]. The resulting tokens comprise the word vocabulary.

The 140 most frequent characters in all pubmed abstracts of year 2015 were chosen as the character vocabulary, accounting for more than 99.5 percent of character appearances.

5.2 Model

For each token, a feature vector is produced as the concatenattion of:

- Word embedding feature: result of look up in a word embedding matrix.
- Capitalization feature with 4 labels: all lowercase letters, at least one uppercase, starting with uppercase, all uppercase. This also indexes a embedding matrix with 4 entries.
- Character-form feature: The last hidden state of a forward and the last hidden state of a backwards RNN operating on the characters of each input word. The inputs to these RNNs are looked up in a character-level embedding matrix. (The hidden state of the character-level RNNs are reset at each word boundary.)
- Abbreviation feature: whether or not the token is part of an abbreviation where the long form is present in the same abstract. Created using [12].

This token feature vector then goes through a drop-out layer [21] and then into a bi-directional RNN. That RNN's hidden state is fed into a softmax layer classifying into BIO ("begin", "in", "out") tags for each entity. For example, an instance of the CHEM class is represented as beginning B-CHEM and as many I-CHEM needed to cover the CHEM entity.

The RNN variant used is the vanilla one with ReLu activation. Per [17] the performance of a properly-initialized basic RNN is often close to the more advanced variants such GRUs.

The goal of the character-level RNN is to capture the morphological information of the words themselves. There is plenty of structure useful for NER especially for GENE and CHEM classes (see examples in section 3). It replaces traditional features such as character N-grams, suffixes, prefixes.

This model is very similar to the current general NER state-of-art models published in [13]-[15]. The code implementation is entirely our own.

The model arhitecture is shown in figures 1, 2. Figure 1 shows the creation of the feature vector for the token "BRACA1" (a well known gene) . The upper part of the model processing for the sentence "BRCA1 mutations cause ovarian carcinomas" is shown in figure 2.

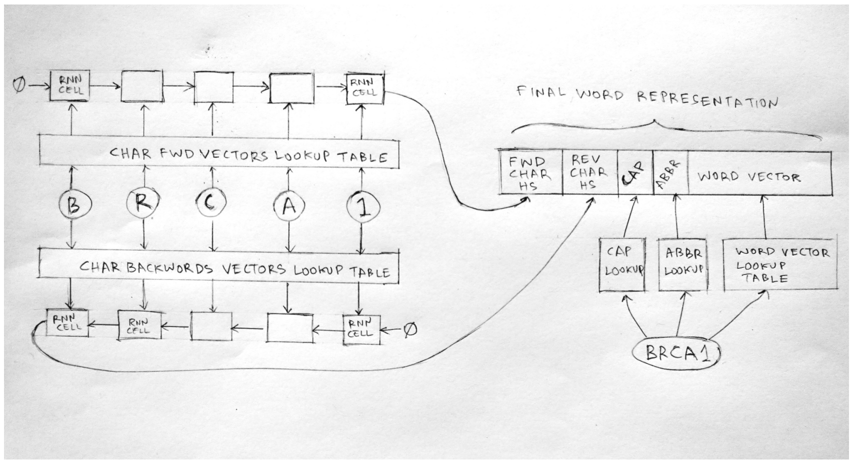

Figure 1: Token feature generation for word "BRCA1"

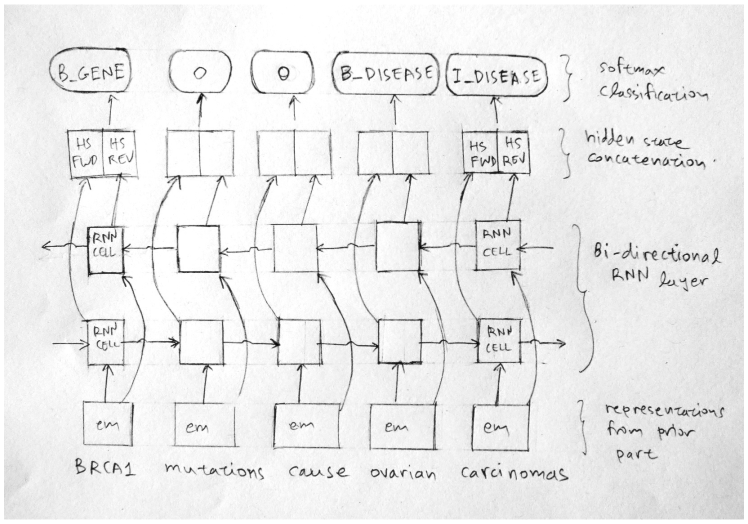

Figure 2: Upper model layers, with example of input sentence and predictions

5.3 Evaluation

We report the "micro"-averaged F-1 measure across all abstracts in the set under evaluation. A predicted entity is a "true positive" if it matches exactly the boundaries and the label of the true entity. This is adversarial for partial matches on long entities, but is consistent with the literature. The benchmark systems are: DNorm [4] for DISEASE, model 1 of tmChem [5] for CHEM in CDR corpus, and for CHEM in CHEMD corpus the winner of the CIM task in [6].

5.4 Training and Results on the Large Noisy Corpus

We downloaded all Pubmed abstracts and the machine-labeled annotations for years 2000 - 2015 for about 12 million abstracts. We created training sets A, B, validation set and test set by sampling randomly 201, 351, 20, 20 thousand abstracts respectively. These subsampled data sets have *zero* overlap. The training sets have 48.6 and 84.8 million tokens respectively.

We trained 100-dimensional word vectors on all abstracts using GloVe [18] for the highest frequency tokens and we used them to initialize the model word embeddings matrix. The recurrent matrices were initialized as scaled diagonal as recommended in [17]. Biases were initialized to 0 and everything else using Xavier initialization [19].

The model was trained using the ADAM optimizer [20]. The best hyperparameters were chosen on the validation set are: learning rate 0.001, drop-out 0.2, top bi-RNN hidden size 2*200 or 2*300, character bi-RNN hidden size 2*25 or 2*50, character embeddings 25. The optimum mini-batch size was found to be surprisingly high: 800 or 1600 tokens in each batch, and with each token's characters going through the character-level RNN on each batch, each gradient update also included 800 or 1600 character-sequences! (All RNN hidden state was reset at document boundaries even when within the same mini-batch.) We experimented with gradient clipping for the recurrent weights and it had no effect or it hurt accuracy when too strong.

Table 1 shows entity-level results for some combinations with 10 epoch training. V, C, H stand for word vocabulary size, hidden state size of each direction of character-level RNN, hidden state size of each direction of top-RNN. The number of training abstracts in thousands is included. The token-level confusion matrix is shown in the Appendix table 7. The baseline loss for uniform at random prediction across 9 classes is 2.197.

Figures 3, 4 show the learning curves. The dotted line is the training loss, the solid the validation loss.

The "elbow points" every 3 epochs are the result of annealing the learning-rate by half which consistently dropped the validation loss when it was flattening out. Figure 3 shows the effect of different

Table 1: Per-epoch training time (minutes), loss, test set entity-level F1-measure

model, training size	time	dev loss	test loss	GENE	DISEASE	CHEM	SPECIES
V50k, none, H200, 201k	90	0.06517	0.06494	76.76	76.69	81.07	95.11
V50k, C25, H200, 201k	164	0.06266	0.06268	78.15	76.95	81.81	95.10
V75k, none, H200, 201k	113	0.06375	0.06351	77.67	76.97	81.71	95.34
V75k, C25, H200, 201k	209	0.06200	0.06184	78.67	77.07	82.20	95.43
V75k, C25, H300, 201k	235	0.06166	0.06149	78.81	77.16	82.50	94.44
V75k, C50, H200, 201k	260	0.06147	0.06121	78.65	77.16	82.39	95.41
V75k, C25, H200, 351k	327	0.05954	0.05940	79.24	78.05	82.53	95.52
V75k, C50, H200, 351k	440	0.05860	0.05833	79.57	78.39	82.78	95.68

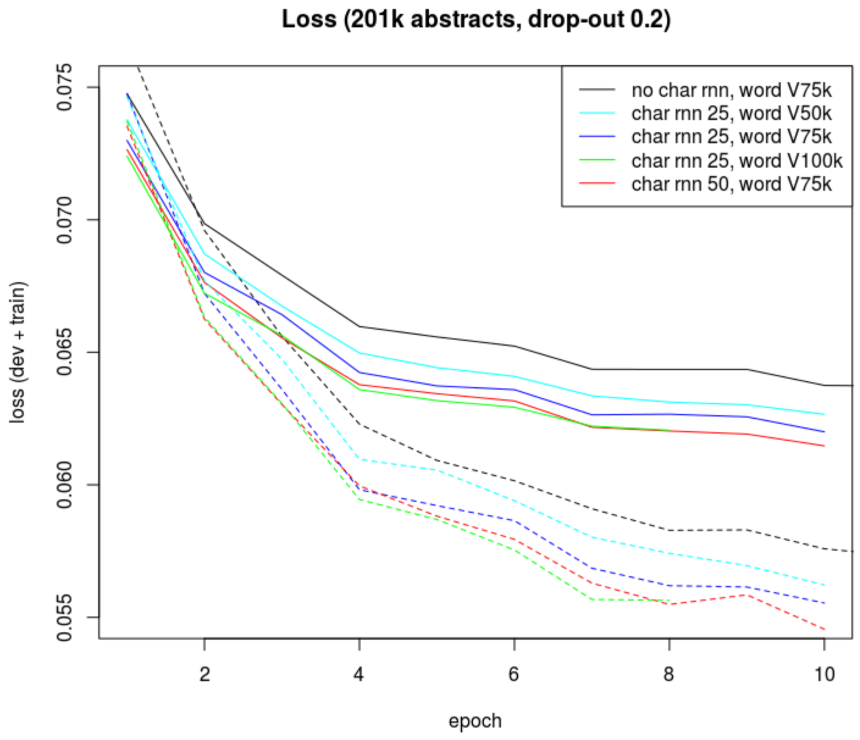

Figure 3: Effect of character-level RNN and word vocabulary size

vocabulary sizes and the character-level RNN. Figure 4 shows that drop-out clearly prevents overfitting and that initializing with pre-trained GloVe word vectors mildly helps. Going from 201,000 training abstracts to 351,000 improves performance as expected. Among variations, the character-level RNN brings the highest improvement. Increasing the character RNN hidden state from 2*25 to 2*50 is worth about as much as increasing the higher layer RNN hidden state from 2*200 to 2*300.

It seems that this type of model has extracted most of what it could from this data set and increasing parameter sizes or using more training data will improve results somewhat but not substantially. By manually inspecting the model predictions when it disagrees with the (noisy) training data, in about one third of the cases the *model is correct instead of its training data*, so we are reaching the limits of the training data itself and start risking fitting its errors harder.

We apply the learned model as-is in the small data sets of section 4.2 and report the results in table 2. In the next session we improve them.

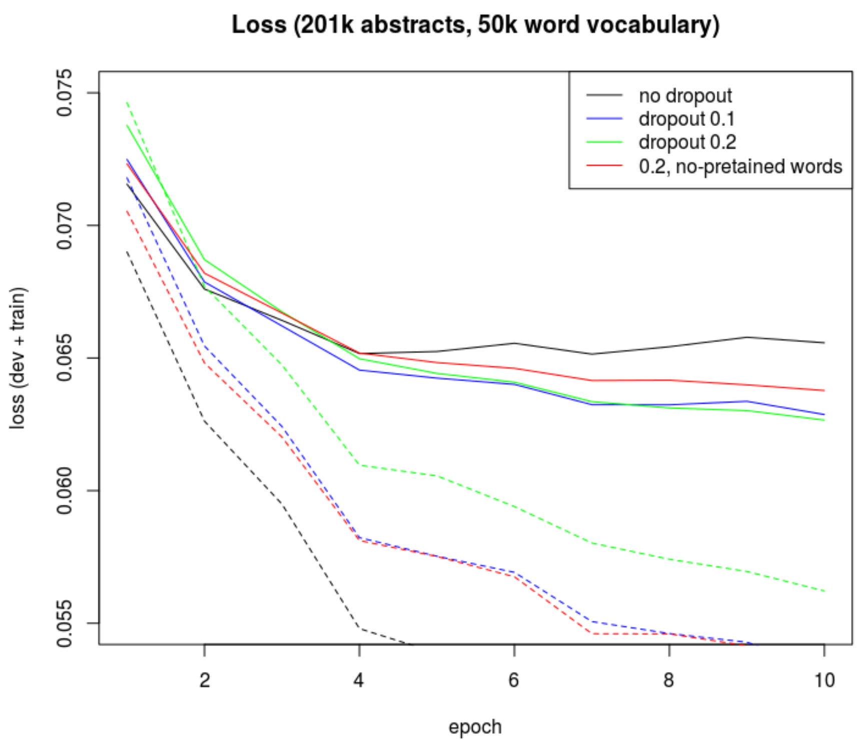

Figure 4: Effect of drop-out and pretrained word vectors

Table 2: Entity-level F1 on test sets of NCBI, CDR, CHEMD Corpora

model, training size	dev loss	NCBI DIS	CDR DIS	CDR CHEM	CHEMD CHEM
V50k, C25, H200, 201k	0.06312	80.21	79.44	84.19	70.91
V75k, none, H200, 201k	0.06375	78.95	79.41	83.67	70.09
V75k, C25, H200, 201k	0.06200	79.79	79.26	84.35	71.30
V75k, C25, H300, 201k	0.06166	78.61	78.90	84.63	71.46
V75k, C50, H200, 201k	0.06147	80.00	79.20	84.84	71.64
V75k, C50, H200, 351k	0.05860	81.51	79.28	84.98	71.83
benchmark	-	79.80	80.70	88.40	88.20

5.5 Training and Results on the Small Human-annotated Corpora

In this section we discuss how to train the model on the small data sets. We start with models "V75k, C25, H200, 351k" and "V75k, C50, H200, 351k" and refer to them as "C25" and "C50" for brevity. They have 7,657,694 and 7,692,244 trainable parameters respectively, prohibitively high for the small data sets. We make the learned word embeddings constant. This reduces the model parameters to 155,188 and 189,738. We slice off in the top softmax layer the classes that do not exist in each small data set and continue training the models on the small data sets. By tuning on the validation set we choose drop-out 0.3 and learning rate 0.0004 and apply the model with the lowest validation loss on the test data.

In order to isolate the effect of the pretrained model, we also create a "baseline" model that is identical to the above except that the trainable parameters are initialized randomly using Xavier initialization [19] and zero biases instead of the pretrained weights. Then that model is trained exactly as above.

The results are shown in tables 3, 4. The column “original” is the model produced in section 5.4 (and after softmax slicing) and “re-trained” is the one produced in this section.

Table 3: Entity-level F1 on NCBI disease corpus

set	C25 baseline	C25 original	C25 re-trained	C50 original	C50 re-trained	benchmark
dev	79.38	76.79	84.05	81.02	87.11	-
test	80.97	80.72	85.34	80.95	86.01	79.80

Table 4: Entity-level F1 on CHEMD chemical corpus

set	C25 baseline	C25 original	C25 re-trained	C50 original	C50 re-trained	benchmark
dev	80.76	73.26	84.19	73.42	84.73	-
test	81.47	73.08	84.69	73.14	85.07	88.20

We see that the predictive ability of model improves dramatically to attain very competitive performance. Moreover the “baseline” model that only differs in initiliazation is far behind. These show that the model indeed captures a lot of information during its long training on the large data set and needs just some fine-tuning to “transfer” that knowledge and unleash its hidden predictive power.

5.6 Error Analysis

Due to space limitations, we report errors for 5.5 only. Errors for 5.4 vary more and also include the very common cases where the lower quality training data is actually incorrect. The most common error for both DISEASE and CHEM classes is exact boundary detection, usually with long entities or when modifier words cause ambiguities. For instance (pubmed id: 8636252), the model identifies entity “childhood kidney cancer” when the true entity is “kidney cancer”, and in another instance (pubmed id: 9272171) it identifies “globoid cell leukodystrophy” when the true entity is “adult onset globoid cell leukodystrophy”. Note that age qualifier is part of the true entity in one case but not in the other, highlighting the challenges of the task. One example from the CHEM class is the text (pubmed id: 23623820): “One new compound, ethyl 13(2) (S)-hydroxy-chlorophyllide a(1)...”. The model outputs entity “ethyl 13(2) (S)-hydroxy-chlorophyllide” but the true entity is “ethyl 13(2) (S)-hydroxy-chlorophyllide a”. Another common error for the CHEM class is with very short all-uppercase identifiers, usually when long-range dependencies are involved. An example is: “Subsequently, zinc oxide and alumina ALD precursors..”. The model correctly identifies “zinc oxide”, and “alumina”, but wrongly identifies “ALD”. In sentence “AGAP2 is a multidomain Arf GAP (ADP ribosylation factor-directed GTPase-activating protein)” the model predicts no CHEM entity, but GAP is one. Finally, words not in the vocabulary occasionally cause errors, but usually are handled well.

6 Final Conclusions

Our goal was to apply a complex state-of-the-art neural network for NER on a specialized domain with tiny public datasets. We follow a semi-supervised learning approach to bootstrap training: We learned networks on lower quality training data. These networks do capture a lot of information from the large corpus, even though the large corpus had noisy labels, but they can’t apply that information directly. However with minimal fine-tuning on the small higher quality data sets they can produce state-of-the-art predictions.

Acknowledgments

The author is greatful to Richard Socher for his guidance on this project. Ignacio Cases, Arun Chaganty and Kevin Clark also provided helpful insights.

References

[1] Lafferty, McCallum, Pereira (2001) Conditional random fields: Probabilistic models for segmenting and labeling sequence data *ICML 2001*

[2] Pubmed / Medline, National Center for Biotechnology Information https://www.ncbi.nlm.nih.gov/pubmed/

[3] Leaman et al. (2008) BANNER: an executable survey of advances in biomedical named entity recognition *Pac Symp Biocomput. 2008:652-63*

[4] Leaman et al. (2013) DNorm: disease name normalization with pairwise learning to rank. *Bioinformatics. 2013 Nov 15;29(22):2909-17*

[5] Leaman et al. (2015) tmChem: a high performance approach for chemical named entity recognition and normalization *Journal of Cheminformatics 2015, 7(Suppl 1):S3*

[6] Krallinger et al. (2015) CHEMDNER: The drugs and chemical names extraction challenge. *Journal of Cheminformatics 2015, 7(Suppl 1):S1*

[7] The NCBI Disease Corpus (2013) https://www.ncbi.nlm.nih.gov/CBBresearch/Dogan/DISEASE/

[8] BioCreative IV CHEMDNER corpus (2013) http://www.biocreative.org/news/corpora/bc-iv-chemdner-corpus/

[9] Leaman et al. (2016) BioCreative V CDR task corpus: a resource for chemical disease relation extraction *Database (Oxford)*

[10] Lu et al. (2015) CHEMDNER system with mixed conditional random fields and multi-scale word clustering *Journal of Cheminformatics 2015, 7(Suppl 1):S4*

[11] Stanford Core NLP http://stanfordnlp.github.io/CoreNLP/tokenize.html

[12] Wilbur et al. (2010) Identifying abbreviation definitions - machine learning with naturally labeled data, *Proceedings of the Ninth ICML 2010, pp. 499-505*

[13] Lample et al. (2016) Neural architectures for named entity recognition *NAACL 2016*

[14] Chiu, Nichols (2015), Named Entity Recognition with Bidirectional LSTM-CNNs

[15] Ma, Hovy (2016) End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF *ACL 2016*

[16] Collobert et al. (2011) Natural language processing (almost) from scratch. *JMLR 12:2493-2537*

[17] Quoc V. Le, Jaitly, Hinton (2015) A Simple Way to Initialize Recurrent Networks of Rectified Linear Units *arXiv 2015*

[18] Pennington, Socher, Manning (2014) GloVe: Global Vectors for Word Representation

[19] Glorot, Bengio (2010) Understanding the difficulty of training deep feedforward neural networks *AISTATS 2010*

[20] Kingma, Ba (2015) ADAM: A Method For Stochastic Optimization *ICLR 2015*

7 Appendix

The following is the token-level confusion matrix produced by of model “V75k, C25, H200, 351k” when applied on the test set (section 5.4)

Table 5: Token-level confusion matrix

Class	O	B-GENE	I-GENE	B-DIS	I-DIS	B-CHEM	I-CHEM	B-SPEC	I-SPEC	
K	0	1	2	3	4	5	6	7	8	Recall
0	4481162	5506	4596	11273	7289	6597	4460	1537	903	99.068
1	5042	30407	53	341	8	321	9	9	0	84.020
2	4977	241	20110	26	94	55	128	2	0	78.454
3	14848	292	24	75731	2389	306	0	579	28	80.396
4	13721	42	95	2606	47237	102	62	152	341	73.397
5	5125	358	33	170	15	36884	170	6	0	86.256
6	2989	3	107	1	27	110	12392	2	0	79.278
7	1147	30	2	400	46	1	1	53618	51	96.965
8	997	3	34	13	154	1	0	42	6390	83.704
Precis.	98.922	82.444	80.267	83.624	82.497	83.115	71.954	95.837	82.847	
F-1	98.995	83.225	79.350	81.979	77.682	84.657	75.439	96.398	83.274	