Swiss Chocolate at CAp 2017 NER Challenge: Partially Annotated Data and Transfer Learning

Nicole Falkner[1], Stefano Dolce[2], Pius von Däniken[3], and Mark Cieliebak[4]

[1]Zurich University of Applied Sciences
[2]Zurich University of Applied Sciences
[3]SpinningBytes AG
[4]Zurich University of Applied Sciences

June 14, 2017

Abstract

In this paper, we describe several deep learning approaches for Named Entity Recognition (NER) for the case where only little annotated data is available. We show that training a separate network per entity greatly improves the precision of a network and transfer learning on a different language or a partially annotated corpus increases the F1-score by up to 7 points. Our transfer learning system was evaluated in the CAp 2017 competition for Named Entity Recognition on French tweets, where it achieved 5th place, obtaining an F1-score of 50.05.

Keywords: Named Entity Recognition, Transfer Learning, Partially Annotated Data, Conceptualisation

1 Introduction

Named Entity Recognition (NER) is an important task in Natural Language Processing. Its goal is to recognise entities such as names of people or locations. NER can be used in many different applications when new entities should be recognised without relying on a rule based system. Today there are state of the art F1-scores of around 92 points on English news texts [CN15]. For more complex domains such as Twitter there is less information available, and in the WNUT 2016 NER competition on English Twitter data the winners achieved an F1-score of 52.41 [STR+16].

In this paper, we propose two different approaches to improve Named Entity Recognition, both using the same basic sequence labelling system described in Section 2.1. The general idea is to use additional data to improve the results. As recent research shows improvements when using transfer learning [YSC17], our main approach is to use transfer learning in combination with the basic sequence labelling system. Furthermore we evaluate the use of automatically generated partially annotated data. Finally, we also add an additional baseline, by training a separate model per entity type as described in Section 2.3.

The performance of the different approaches is evaluated using the French Twitter dataset of the CAp 2017 NER competition [LPB+17].

2 Model

2.1 Basic Sequence Labelling System

Network Our baseline system is a bidirectional Long Short-Term Memory RNN [HS97] as previously used in [CN15], which achieved state-of-the-art performance for Named Entity Recognition on the English CoNLL 2003 [TKSDM03] corpus. Figure 1 shows an overview of the system. For every word w_i in a given sequence of words $w_{1:T}$, we first compute a feature vector, which is the concatenation of all the features described in Section 2.4. The resulting sequence of feature vectors $v_{1:T}$ is then fed to a bidirectional LSTM. The output of both the forward and backward LSTM are concatenated to get $o_{1:T}$, which get passed through a *ReLU* activation function. We apply dropout [SHK+14] of 0.3 to $o_{1:T}$. To get the final tag probabilities, we pass $o_{1:T}$ through a fully connected layer with softmax activation.

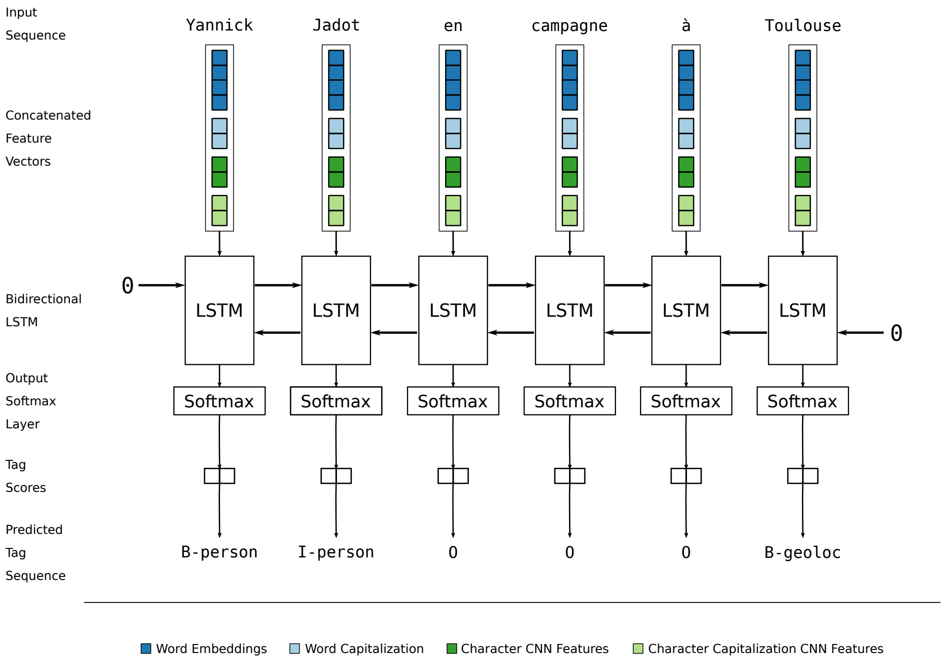

Figure 1: BLSTM-CNN, baseline bidirectional LSTM model as in [CN15, CWB+11]

Training and Inference Since there are additional restrictions on valid tag sequences (e.g. in BIO2 encoding an I-person tag cannot be followed by an I-product tag), we employ the same strategy as described in [CN15, CWB+11]. We use a matrix A, such that $A_{i,j}$ is the score of transitioning from tag i to tag j. The $A_{i,j}$ are initialized from a random uniform distribution over the interval $\left[-\frac{1}{2}, \frac{1}{2}\right]$.

Let θ be the parameters of the network described above. Let $f_{\theta,t,i}(w_{1:T})$ be the score the network parametrized by θ outputs for tag i at time step $1 \leq t \leq T$ for the input sequence $w_{1:T}$. The score of a given sequence of tags $i_{1:T}$ is then computed as the sum of the network and transitions scores:

$$S\left(i_{1:T}; w_{1:T}, A, \theta\right) = \sum_{t=1}^{T} \left(A_{i_{t-1},i_t} + f_{\theta,i_t,t}\left(w_{1:T}\right)\right)$$

We then compute the log-likelihood of the true label sequence $y_{1:T}$ by computing the softmax over all possible label sequences:

$$\begin{aligned} &\log P\left(y_{1:T} | w_{1:T}, A, \theta\right) = \\ &\quad S\left(y_{1:T}; w_{1:T}, A, \theta\right) - \log \sum_{\forall j_{1:T}} \exp\left(S\left(j_{1:T}; w_{1:T}, A, \theta\right)\right) \end{aligned}$$

This can be computed efficiently using dynamic programming. At inference time, we can use the Viterbi algorithm to find the sequence with maximal score [CWB+11, CN15].

We use the Adam [KB14] algorithm to optimize the weights θ and A, with a learning rate of 0.003. The exponential decay rate for first moment estimates is set to 0.9 and to 0.999 for second moment estimates. Additionally, we perform gradient clipping [PMB12] with norm constraint 10. For all experiments, we train the systems for 10 epochs without early stopping.

2.2 Transfer Learning Architecture

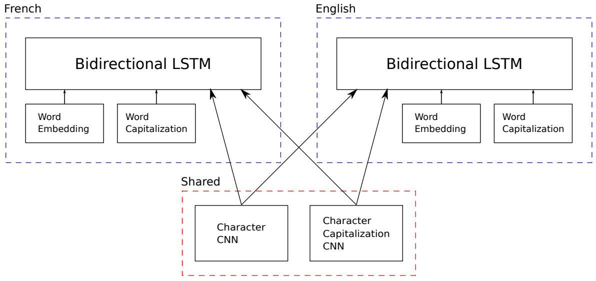

Figure 2: Overview of separated and shared network weights of our transfer learning architecture.

Our transfer learning approach is motivated by the existence of a similar English corpus, WNUT 2016, which is also based on Twitter data and has a large overlap in entity types. Moreover, English and French have a considerable amount of cognates, and we hope to exploit this when transferring character level features.

In order to augment the French data set with additional English training data, we use the cross-lingual transfer model proposed in [YSC17]. We use separate basic models, see Section 2.1, for both languages but allow the weights for the character and character capitalization CNNs to be shared between them, see Figure 2. During training, we choose training samples i.i.d from each language to update the language specific and shared weights.

2.3 Separate Networks per Entity Type

As described in Section 3.1, we augment the data set with partially annotated data. We want to study how additional noisy data influences different entity types separately, therefore we train a separate version of the system in Section 2.1 for every entity type and aggregate the results.

2.4 Features

Word Embeddings We use a word2vec model as described in [MSC+13] to compute word embeddings. For both French and English the skip-gram context window length is 5 and the embedding dimension is 200. For each language, the word2vec model is trained on a corpus of 200M Twitter messages. This results in an embedding matrix $E_{word} \in \mathbb{R}^{n_l+1\times 200}$ where n_l is the number of tokens in the vocabulary for language l. We add an additional vector, initialized from a zero mean and unit variance Gaussian, for unknown words and padding.

Word Capitalization Features Following [CN15, CWB+11] we add explicit capitalization features, since capitalization information is lost during the word embedding lookup. The feature options are: all capitalized, uppercase initial, all lower cased, mixed capitalization, and other. An embedding matrix $E_{wordCap} \in \mathbb{R}^{5\times d_{wordCap}}$ is used to feed the features to the network. $E_{wordCap}$ is initialized randomly from a Gaussian distribution with mean zero and unit standard deviation and updated during training via backpropagation. $d_{wordCap}$ was set to 5 for all experiments.

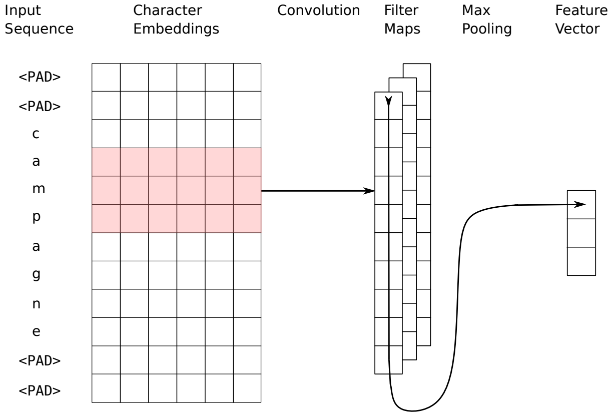

Figure 3: Architecture of the Neural Network used to extract character level features.

Character Convolution Features A convolutional neural network is used to extract additional character level features. An overview is shown in Figure 3. First, we add special padding tokens on both sides of the character sequence w, to extend it to a target maximum length, $l_{w,max}$. In the case where the number of paddings is odd, the additional padding token is added on the right. If $w > l_{w,max}$ only the first $l_{w,max}$ characters are used. An embedding matrix $E_{char} \in \mathbb{R}^{N_c\times d_c}$ maps characters to $\mathbb{R}^{d_c}$ vectors, where N_c is the cardinality of the set of characters considered, including the special padding token and a separate token for unknown characters. The elements of E_{char} are initialized from uniform distribution over the interval $\left[-\frac{1}{2}, \frac{1}{2}\right]$. The value of d_c is set to 15 for all experiments.

Using E_{char} we embed the whole sequence w and obtain $C_w \in \mathbb{R}^{l_{w,max}\times d_c}$. A set of m convolution filters $\in \mathbb{R}^{d_c\times h}$ is then applied to C_w. No additional padding is applied to C_w and therefore this operation results in m feature maps $M_i \in \mathbb{R}^{l_{w,max}-h+1}$. The M_i are passed through a *ReLU* activation. The final feature vector $F \in \mathbb{R}^m$ is attained by max pooling, $F_i = \max M_i$. In our experiments we fixed $m = 10$ and $h = 3$.

Character Capitalization Convolution Features Analogous to the word capitalization features, we use additional character capitalization features. The feature options are: upper, lower, punctuation, numeric and other. We then apply a neural network with the same convolutional architecture as described above to extract the final character capitalization features for a given word. The parameters are initialized as above, except the embedding dimension which is reduced to 5.

3 Experiments

3.1 Data

The training data provided by CAp 2017 [LPB$^+$17] consists of 3000 French tweets annotated with the following 13 entities: *person, geoloc, org, media, event, movie, tvshow, transportline, product, musicartist, sportsteam, facility* and *other*. For Transfer Learning from English, we used the annotated dataset from WNUT16 [STR$^+$16], as the entities were similar, although not completely the same. The *event, media*, and *transportline* entities were not part of the English data set.

Partially Annotated Data To generate partially annotated data we download lists of a few thousand named entity instances from Wikidata[1] for each entity except *transportline* and *other*, since those were not readily available in the Wikidata corpus. Using these lists we search for around 900'000 tweets from Twitter. To tag these tweets, we use a conceptualization approach as described in [KWO13]. The basic idea is that a named entity can be associated with different concepts. For example "apple" can have the concept of "fruit" or "company" with different probabilities. To determine the probabilities we use the Concept Graph API which is based on the Probase Knowledge Base [WW16, WWWX15, WZW$^+$15, HWW$^+$15, WWH14, SWW$^+$11]. To retrieve possible concepts we randomly sample 500 named entity instances from each retrieved list. Instances that did not return a concept were not taken into account. If more than 5 instances returned the same concept we additionally download 200 tweets per concept. Using the concept and the entity tweets, applying a stop word filter and a Snowball Stemmer, we train an LDA model with 50 topics [BNJL03]. We then tag the entities in the instance list tweets using the procedure visible in Figure 4.

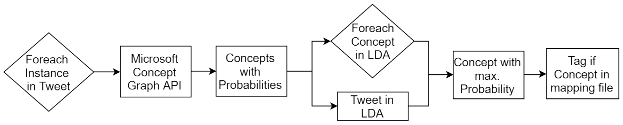
Foreach Instance in Tweet
Microsoft Concept Graph API
Concepts with Probabilities
Foreach Concept in LDA
Tweet in LDA
Concept with max. Probability
Tag if Concept in mapping file

Figure 4: Overview how additional tweets are tagged

For the likelihood of a concept we translate a found entity to English and send it to the Microsoft Graph API. To apply the concept to the French LDA we then translate the concept to French. This allows us to find the probability of the concept given our LDA model. We manually map the concepts to the entities which we want to tag.

We use the approach proposed in [KWO13] to find the most likely concept c given a word w in a sentence s.

$$p\left(c|w,s\right) = \alpha p\left(c|w\right) \sum_{t=1}^{k} p\left(t|c\right) p\left(t|s\right)$$

$p\left(c|w\right)$ is given by the Microsoft Graph API, $p\left(t|c\right)$ and $p\left(t|s\right)$ are given by the topic model and α is a scaling factor. We chose $\alpha = \frac{1}{1000}$ and the number of topics $k = 50$.

When looking at the tagged data we realized that it had a weakness of often tagging too much. For example for movies, there are instances such as "Je" or "1984". If a sentence is about another movie but also contains those two words, all three instances are tagged as a movie. All experiments using partially annotated data were run on data which contain this weakness, assuming that the large amount of data will level these mistakes.

3.2 Evaluated Systems

In the experiments we compare the three different architectures using annotated or partially annotated data. We will refer to these as:

- **BLSTM-CNN**: The basic bidirectional LSTM combined with CNN for character level features as reference for the proposed systems (Section 2.1)
- **BLSTM-CNN-PAD-M**: BLSTM-CNN (Section 2.1) with partially annotated data trained in all epochs
- **BLSTM-CNN-PAD-P**: BLSTM-CNN (Section 2.1) using partially annotated data only for pre-training
- **S-BLSTM-CNN-PAD-P**: Separate BLSTM-CNN per entity (Section 2.3) using partially annotated data only for pre-training
- **TL-BLSTM-CNN-EN**: BLSTM-CNN with shared Character embeddings (Section 2.2) using the English WNUT 2016 dataset to train the source system
- **TL-BLSTM-CNN-PAD**: BLSTM-CNN with shared Character embeddings (Section 2.2) using the partially annotated data described in 3.1 to train the source system

We refer to pre-training as first training the system on the partially annotated data for one Epoch before moving on to the training with the annotated data.

[1] `https://query.wikidata.org/`

System	Precision	Recall	F1	Accuracy
BLSTM-CNN	45.91%	34.39%	39.30	94.60%
BLSTM-CNN-PAD-M	27.95%	10.31%	15.12	92.54%
BLSTM-CNN-PAD-P	49.96%	31.59%	38.68	94.57%
S-BLSTM-CNN-PAD-P	**80.94%**	29.04%	44.58	**95.40%**
TL-BLSTM-CNN-EN	53.95%	**39.34%**	**45.71**	95.14%
TL-BLSTM-CNN-PAD	50.63%	39.22%	44.18	95.01%

Table 1: Experimental Results

3.3 Results

Table 1 shows the average results from 10 test runs obtained by the systems on the CAp 2017 competition test dataset. The results show that using the partially annotated data to fully train the system makes it worse, and also simply pre-training it shows a slight decrease of the F1-score. Training a separate network per entity shows a significant increase in the precision of the system and thus allows the system to achieve a higher F1-score at the cost of some recall. We see that transfer learning on a similar domain and language can improve the F1-score by about 7 points. An interesting aspect for further research is that partially annotated data combined with transfer learning is almost as good as transfer learning from a fully annotated corpus.

Based on these results we decided to submit the transfer learning with English as source language for the CAp 2017 competition and achieved 5th place with an F1-score of 50.05. The score is 2 points behind the second place score and 9 points behind the winner.

4 Conclusion

We described several deep learning approaches for Named Entity Recognition on little annotated data. Partially annotating data as additional input can improve the results slightly, but does not provide as much improvement as using it for transfer learning or even transfer learning from another language. Training a separate network for each entity increases the precision of the system when used in combination with partially annotated data. Our cross-language transfer learning approach was evaluated on the CAp 2017 competition for Named Entity Recognition and achieved an F1-score of 50.05. For future work we will improve the tagging of partially annotated data and combine the different approaches. Additionally we plan to study the influence of different hyperparameter settings for all subsystems in more detail.

References

[BNJL03] David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003, 2003.

[CN15] J. P. C. Chiu and E. Nichols. Named Entity Recognition with Bidirectional LSTM-CNNs. *ArXiv e-prints*, November 2015.

[CWB+11] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[HWW+15] Wen Hua, Zhongyuan Wang, Haixun Wang, Kai Zheng, and Xiaofang Zhou. Short Text Understanding Through Lexical-Semantic Analysis. In *International Conference on Data Engineering (ICDE)*, April 2015.

[KB14] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *ArXiv e-prints*, December 2014.

[KWO13] Dongwoo Kim, Haixun Wang, and Alice Oh. Context-dependent conceptualization. In *IJCAI*, pages 2654–2661, 2013.

[LPB+17] Cédric Lopez, Ioannis Partalas, Georgios Balikas, Nadia Derbas, Amélie Martin, Coralie Reutenauer, Frédérique Segond, and Massih-Reza Amini. French named entity recognition in twitter challenge. Technical report, 2017.

[MSC^{+}13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, NIPS'13, pages 3111–3119, USA, 2013. Curran Associates Inc.

[PMB12] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training Recurrent Neural Networks. *ArXiv e-prints*, November 2012.

[SHK^{+}14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014.

[STR^{+}16] Benjamin Strauss, Bethany E. Toma, Alan Ritter, Marie-Catherine de Marneffe, and Wei Xu. Results of the WNUT16 Named Entity Recognition Shared Task. In *The 2nd Workshop on Noisy User-generated Text*, pages 138–144, 2016.

[SWW^{+}11] Yangqiu Song, Haixun Wang, Zhongyuan Wang, Hongsong Li, and Weizhu Chen. Short Text Conceptualization Using a Probabilistic Knowledgebase. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Three*, IJCAI'11, pages 2330–2336. AAAI Press, 2011.

[TKSDM03] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada, 2003.

[WW16] Zhongyuan Wang and Haixun Wang. Understanding Short Texts. In *The Association for Computational Linguistics (ACL) (Tutorial)*, August 2016.

[WWH14] Zhongyuan Wang, Haixun Wang, and Zhirui Hu. Head, Modifier, and Constraint Detection in Short Texts. In *International Conference on Data Engineering (ICDE)*, January 2014.

[WWWX15] Zhongyuan Wang, Haixun Wang, Ji-Rong Wen, and Yanghua Xiao. An Inference Approach to Basic Level of Categorization. In *ACM International Conference on Information and Knowledge Management (CIKM)*. ACM – Association for Computing Machinery, October 2015.

[WZW^{+}15] Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. Query Understanding through Knowledge-Based Conceptualization. In *International Joint Conference on Artificial Intelligence (IJCAI)*, July 2015.

[YSC17] Z. Yang, R. Salakhutdinov, and W. W. Cohen. Transfer Learning for Sequence Tagging with Hierarchical Recurrent Networks. *ArXiv e-prints*, March 2017.