Knowledge Transfer in Neural Language Models

Peter John Hampton, Hui Wang, and Zhiwei Lin

Artificial Intelligence Research Group
Jordanstown, Antrim, Ulster University, BT37 0QB
{hampton-p1,h.wang,z.lin}@ulster.ac.uk
https://ulster.ac.uk

Abstract. The complexity and depth of Information Extraction becomes increasingly apparent as time goes on. Heuristics, Statistical Learning and more recently, Neural Models have proved challenging to scale into, and out of various domains. In this paper we discuss the limitations of current approaches and explore if transferring human knowledge into a neural language model could improve performance in an Information Extraction setting. We approach this by constructing gazetteers from existing public resources and add to a deep recurrent topology with a shallow classifier output. We demonstrate that leveraging existing knowledge we can increase performance and train the networks faster. We argue a case for further research into leveraging pre-existing domain knowledge and engineering resources to train such models.

Keywords: Hybrid Systems, Information Extraction

1 Introduction

In 2016, Natural Language Processing has been dubbed a *rabbit in the headlights of Deep Learning* [1]. It is certainly plausible given the recent achievements of Deep Learning in the image classification and object recognition space [2, 3]. In addition, Open Source communities have worked to commoditize Deep Learning capabilities through high-level frameworks thus lowering the barrier to entry for new research and Greenfield projects [4, 5]. In this vast growing research field, it is of interest to revisit research from previous years and look to how systems can be potentially improved with representation learning.

One space with an abundance of research and applications is the field of Information Extraction (IE). IE is regarded as an umbrella term for classification tasks such as Named Entity Recognition, Relation Extraction and Coreference Resolution but is typically used in other areas such as Information Retrieval (Entity Search), stylometry, vocabulary analysis and so on [6]. This paper focuses on established research in the areas of Named Entity Recognition applying a Bidirectional Long-Short Term Memory (LSTM) with two shallow classifiers to the CoNLL 2003 shared task [7]. Leveraging open gazetteers with word embeddings we achieve performance greater than shallow methods and those with stringent heuristics. We conclude this preliminary study by discussing future directions for domain-specific deep learning based entity mining.

2 Information Extraction

It would be rational to assume true understanding of language requires consciousness similar to that experienced by human beings. Language not only evolves over time, it arguably evolves in ones mind based on their interactions with the world. The longevity of Information Extraction tasks has been sustained due to the exponential growth in unstructured content in previous years. Various languages, document genres, entity types and knowledge connections have interested researchers and practitioners in this area [8].

It still common to find researchers and engineers making use of rules or gazetteers when identifying entities. This works well in narrow cases, but this approach alone is often unscalable, deterministic and doesn't work well with the ambiguous nature of natural languages. Researchers have popularized stochastic methods such as Markovian Classification Models and Conditional Random Fields (the current state of the art) to classify named entities. Some work aims to combine the advantages of the two to reduce implementation risk. This works similar to a typical classification problem for a set $D = \{(x_i, y_i)\}_{i=1}^{N}$ of N words, where y_i is the label for word x_i except a sequence of vectors is given $(x_1, x_2, ..., x_n)$ and returns some information relative to the input sequence $(y_1, y_2, ..., y_n)$.

Attention has been focused on activations, model optimization, training networks and architectures and learning from data with no explicit feature engineering, thus reflecting the real world. These architectures include, but not limited to, Convolutional Neural Networks (ConvNets), Recurrent Neural Networks (RNNs), and Gated Recurrent Units (GRUs). In this study, we focus on a Hybrid LSTM-CRF leveraging gazetteers built from open IE (ANNIE) data.

2.1 Data

CoNLL For the experiments we use the CoNLL 2003 English dataset. This popular dataset is split into three parts: Training (train), Validation (testa) and Testing (testb). The data set is split at a document, sentence, and word level. The data has two features included: Part of Speech and Chunk Tags.

Gazetteers Gazetteers are a common input format for rule based methods. It's not uncommon to see applications leverage them as organizations tend to be abundant in columnar data. They are employed in various studies with often random results. This is because there is no predefined agreed upon set of gazetteers. It's possible to create *task specific* gazetteers. Although these types of gazetteers produce great results in various IE tasks, they don't generalize well to external unseen data. This is why we create the gazetteers used in this study from the ANNIE system as described by [11].

Word Embeddings Word embeddings are the result of mapping semantic meaning into a highly dimensional geometric space. This is done by associating a numeric vector to every word in a dictionary. In this study we use the

Glove Pretrained Word Embeddings with 100 dimensions that were trained on an English Wikipedia Dump [10].

3 Methodology

3.1 Long-Short Term Memory Network

Long-Short Term Memory (LSTM) Networks are a type of training method to overcome the general Recurrent Neural Networks bias for more recent inputs and help learn long-term dependencies. Recurrent Neural Networks are networks whose connections between units form a directed cycle which in turn make it good when working with sequential data. This is achieved by introducing memory cells and implementing gates to control the proportion of the input to given and what proportion of the previous state to forget, in essence carrying memory forward $h_t := \theta(Wx_t + Uh_{t-1})$. Where h_t is the hidden state at that time step and $\theta(.)$ is an activation function that is applied to the sum of the weight input W and hidden states U.

Although there are other interesting hybrid variations of LSTMs with convolutional layers and other exotic variations, one promising approach which we have based our initial implementation is the work of [9] who designed the following:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{1}$$

$$c_t = (1 - i_t) \cdot c_{t-1} + i_t \cdot \tanh(W_{xs}x_t + W_{hc}h_{t-1} + b_c) \tag{2}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{3}$$

$$h_t = o_t \cdot \tanh(c_t) \tag{4}$$

where σ is an element-wise sigmoid function. When training in both directions (with different parameters) the word representations are learned by concatenating the right and left context representations, $h_t = [h_t\rightarrow; h_t\leftarrow]$. Another interesting product of their research was adoption of the IOBES tagging scheme, an evolution of IOB (Inner-Outer-Beginning) which marks entities (s) and the end of entities (e). However, they claim their early work didn't yield any noteworthy improvements.

3.2 Conditional Random Fields

Conditional Random Fields (CRFs) have remained the state of the art for quite some time in the Named Entity Recognition space and are good for encoding known relationships between observations and construct consistent interpretations. The conditional probability of a state sequence $X = (x_1, x_2, ..., x_n)$ given some form of observation sequence $Y = (y_1, y_2, ..., y_n)$ is

$$P(X|Y) = \frac{1}{Z_o} exp\left(\sum_{t=1}^{T}\sum_{k}\lambda_k f_k(X_{t-1}, X, Y, t)\right) \tag{5}$$

where $f_k(X_{t-1}, X, Y, t)$ is a feature function whose weight λ_k is to be learned in the training process with Z_o as a normalization function (6). These models define the conditional probability of a label sequence based on total probability over the state sequences (7):

$$Z_o = \sum_{x} \sum_{t=1}^{T} \lambda_k f_k(x_{t-1}, x, y, t) \tag{6}$$

$$P(1|Y) = \sum_{x:1(X)=1} P(X|Y) \tag{7}$$

4 Experiments and Results

4.1 Preprocessing

After loading the CoNLL data, we convert all numbers to '0' and lowercase all the words. We also move off the IOB format for the Chunk and NER tags and convert them to IOBES tags to make the compositionality more deducible. We further normalize the Part of Speech tags by removing any invalid alphabetical based tags.

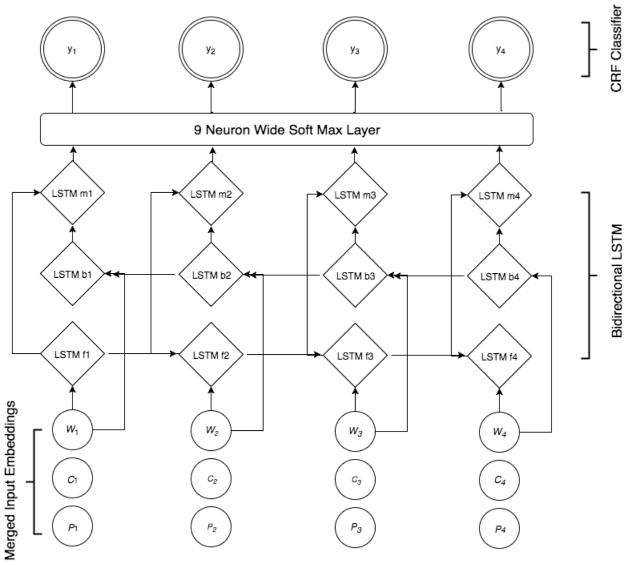
When analysing the training, validation and testing data we found that an average of 96% of Named Entities had the chunk tag I-NP and an average of 90% of the POS tags were NNP. We therefore expanded the gazetteers by labelling all of the words with these Chunk and POS tags and generating tuples like the CoNLL datasets. We finish by concatenating the processed gazetteers onto the training data and then training the model.

4.2 Topology

In our experiment we implement a LSTM (example depiction Fig. 1) network with a softmax output layer that feeds into a CRF classifier. The 3 vectors in the embedding layer are merged together and passed to $LSTMF$ and $LSTMB$. $LSTMF_n$ represents the LSTM training forward and $LSTMB_n$ represents the LSTM training backwards. The $LSTMM_n$ layer is the output of $LSTMF_n$ and $LSTMB_n$ merged together before they are passed to the softmax layer. The number of neurons in the softmax layer is relational to the number of position output (y_n) values. The output of the softmax layer is then passed to the CRF which outputs an index corresponding to a Named Entity tag.

A dropout rate of 0.4 is applied between the BiLSTM and the Softmax layer to prevent overfitting. Each LSTM has 100 units and we have a batch size of 64. A RMSProp optimizer with a learning rate of 0.005 was adopted as it trains the model fast and we found it to generate the best results.

Fig. 1. A visual depiction of our employed network topology



4.3 Results

Our results are presented in Table 1. After running each model for 5 epochs we uncovered two things:

1. A performance gain of 2.25% FB1 was observed
2. The network trained much faster

Table 1. Results of our Experiments compared to the baseline

Model	**F1**
Proposed (No Gaz)	85.73
Proposed (inc Gaz)	**87.98**
LSTM-CRF [9][1]	83.63
Stacked LSTM [9][2]	80.88

5 Conclusions and Future Work

It's clear from recent publications that Deep Learning is becoming the preferential choice in IE research. We argued that there is an abundance of data that researchers and practitioners could leverage to train such models. We demonstrate promising performance gains by creating gazetteers out of the general ANNIE gazetteers and adding it to the existing training set. In the near future we plan to:

1. Implement a separate feed forward network to learn the gazetteer / Named Entity relations and pass it to our existing topology in Figure 1, merging with the Word, Chunk and Part of Speech tuple. We believe this would enable an additional feature gain.
2. Enhance the utilization of gazetteers by generating or retrieving sentences containing the entries. We would ditch the naive POS Chunk labelling and classify the feature tags with existing tools. This should give greater context between words in the context and allow the network to disambiguate the ambiguous terms.

References

1. Manning, C.D., 2016. Computational linguistics and deep learning. Computational Linguistics.
2. LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. Nature, 521(7553), pp.436-444.
3. Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural Networks, 61, pp.85-117.
4. Bahrampour, S., Ramakrishnan, N., Schott, L. and Shah, M., 2015. Comparative Study of Caffe, Neon, Theano, and Torch for Deep Learning. arXiv preprint arXiv:1511.06435.
5. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), pp.2825-2830.
6. Schmitz, M., Bart, R., Soderland, S. and Etzioni, O., 2012, July. Open language learning for information extraction. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (pp. 523-534). Association for Computational Linguistics.
7. Tjong Kim Sang, E.F. and De Meulder, F., 2003, May. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4 (pp. 142-147). Association for Computational Linguistics.
8. Sarawagi, S., 2008. Information extraction. Foundations and trends in databases, 1(3), pp.261-377.
9. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. and Dyer, C., 2016. Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360.
10. Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In EMNLP (Vol. 14, pp. 1532-1543).
11. Bontcheva, K., Derczynski, L., Funk, A., Greenwood, M. A., Maynard, D., & Aswani, N. (2013, September). TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. In RANLP (pp. 83-90).