UTA_DLNLP at SemEval-2016 Task 12: Deep Learning Based Natural Language Processing System for Clinical Information Identification from Clinical Notes and Pathology Reports

Peng Li
Computer Science and Engineering
University of Texas at Arlington
jerryli1981@gmail.com

Heng Huang*
Computer Science and Engineering
University of Texas at Arlington
heng@uta.edu

Abstract

We propose a deep neural network based natural language processing system for clinical information (such as time information, event spans, and their attributes) extraction from raw clinical notes and pathology reports. Our approach uses the context words and their part-of-speech tags and shape information as features. We utilize the temporal (1D) convolution neural network to learn the hidden feature representations. In prediction step, we use the Multilayer Perceptron (MLP) to predict event spans. The empirical evaluation demonstrates that our approach significantly outperforms baseline methods.

1 Introduction

In past several years, there has been much interest in applying neural network based deep learning techniques to solve many natural language processing (NLP) tasks. From low-level tasks such as language modeling, POS tagging, named entity recognition, and semantic role labeling (Collobert et al., 2011; Mikolov et al., 2013), to high-level tasks such as machine translation, information retrieval, semantic analysis (Kalchbrenner and Blunsom, 2013; Socher et al., 2011a; Tai et al., 2015) and sentence relation modeling tasks such as paraphrase identification and question answering (Socher et al., 2011b; Iyyer et al., 2014; Yin and Schutze, 2015). Deep representation learning has demonstrated its importance for these tasks. All the tasks get performance improvement via learning either word level representations or sentence level representations.

In this work, we introduce the deep representation learning technologies to the electronic medical record research. Specifically, we focus on clinical information extraction, using clinical notes and pathology reports from the Mayo Clinic. Our system is designed to identify event expressions consisting of the following components:

- The spans (character offsets) of the expression in the raw text
- Contextual Modality: ACTUAL, HYPOTHETICAL, HEDGED or GENERIC
- Degree: MOST, LITTLE or N/A
- Polarity: POS or NEG
- Type: ASPECTUAL, EVIDENTIAL or N/A

The input of our system consists of raw clinical notes or pathology reports. The following is an example:

> *April 23, 2014: The patient did not have any postoperative bleeding so we will resume chemotherapy with a larger bolus on Friday even if there is slight nausea.*

The output annotations over the text capture the key information such as event mentions and attributes. Table 1 illustrates the output of clinical information extraction in details.

To solve this task, the major challenge is how to precisely identify the spans (character offsets) of

To whom all correspondence should be addressed. This work was partially supported by NSF-IIS 1117965, NSF-IIS 1302675, NSF-IIS 1344152, NSF-DBI 1356628, NIH R01 AG049371.

Proceedings of SemEval-2016, pages 1159–1164,
San Diego, California, June 16-17, 2016.

Clinical Note	Event Mention	Event Attribute
April 23, 2014: The patient did not have any postoperative bleeding so we will resume chemotherapy with a larger bolus on Friday even if there is slight nausea.	bleeding	type=N/A polarity=NEG degree=N/A modality=ACTUAL
	resume	type=ASPECTUAL polarity=POS degree: N/A modality=ACTUAL
	chemotherapy	type=ASPECTUAL polarity=POS degree=N/A modality=ACTUAL
	bolus	type=ASPECTUAL polarity=POS degree=N/A modality=ACTUAL
	nausea	type=ASPECTUAL polarity=POS degree=N/A modality=HYPOTHETICAL

Table 1: An example of information extraction from clinical note.

the event expressions from raw clinical notes. Traditional machine learning approaches usually build a supervised classifier with features generated by the Apache clinical Text Analysis and Knowledge Extraction System (cTAKES) [1]. For example, BluLab system (Velupillai et al., 2015) extracted morphological (lemma), lexical (token), and syntactic (part-of-speech) features encoded from cTAKES. Although using the domain specific information extraction tools can improve the performance, learning how to use this software well for clinical domain feature engineering is still very time-consuming. In short, a simple and effective method that only leverages basic NLP modules and achieves high extraction performance is desired for regular users.

To address this challenge, we propose a deep neural networks based method, especially convolution neural network (Collobert et al., 2011), to learn hidden feature representations directly from raw clinical notes. More specifically, one method first extracts a window of surrounding words for the candidate word. Then, we attach each word with their part-of-speech tag and shape information as extra features. After that, our system deploys a temporal convolution neural network to learn hidden feature representations. Finally, our system uses Multilayer Perceptron (MLP) to predict event spans. Note that we use the same model to predict event attributes.

[1]Apache cTAKES is a natural language processing system for extraction of information from electronic medical record clinical free-text

2 Constructing High Quality Training Dataset

The major advantage of our system is that we only leverage NLTK [2] tokenization and a POS tagger to preprocess our training dataset. When implementing our neural network based clinical information extraction system, we found it is not easy to construct high quality training data due to the noisy format of clinical notes. Choosing the proper tokenizer is quite important for span identification. After conducting experiments, we found that "RegexpTokenizer" can match our needs. This tokenizer can generate spans for each token via sophisticated regular expression such as:

```
nltk.tokenize.RegexpTokenizer
("\w+|\$[\d\.]+|\S+")
```

We then use "PerceptronTagger" as our part-of-speech tagger due to its fast tagging speed. Note that when extracting context words, please make sure you deploy the same tokenization module instead of just splitting strings by space.

3 Neural Network Classifier

Event span identification is the task of extracting character offsets of the expression in raw clinical notes. This subtask is quite important due to the fact that the event span identification accuracy will affect the accuracy of attribute identification. We first run our neural network classifier to identify event spans. Then, given each span, our system tries to identify attribute values.

[2]http://www.nltk.org

3.1 Temporal Convolutional Neural Network

The way we use temporal convolution neural network for event span and attribute classification is similar with the approach proposed by (Collobert et al., 2011). Generally speaking, we consider a word as represented by K discrete features $w \in D^1 \times \cdots \times D^K$, where D^K is the dictionary for the k^{th} feature. In our scenario, we just use three features such as token mention, pos tag, and word shape. Note that word shape features are used to represent the abstract letter pattern of the word by mapping lowercase letters to "x", upper-case to "X", numbers to "d", and retaining punctuation. We associate to each feature a lookup table. Given a word, a feature vector is then obtained by concatenating all lookup table outputs. Then a clinical snippet is transformed into a word embedding matrix. The matrix can be fed to further 1-dimension convolutional neural network and max pooling layers. Below we will briefly introduce core concepts of Convolutional Neural Network (CNN).

Temporal Convolution

Temporal Convolution applies one-dimensional convolution over the input sequence. The one-dimensional convolution is an operation between a vector of weights $\mathbf{m} \in \mathbb{R}^m$ and a vector of inputs viewed as a sequence $\mathbf{x} \in \mathbb{R}^n$. The vector $\mathbf{m}$ is the *filter* of the convolution. Concretely, we think of $\mathbf{x}$ as the input sentence and $\mathbf{x}_i \in \mathbb{R}$ as a single feature value associated with the i-th word in the sentence. The idea behind the one-dimensional convolution is to take the dot product of the vector $\mathbf{m}$ with each m-gram in the sentence $\mathbf{x}$ to obtain another sequence $\mathbf{c}$:

$$\mathbf{c}_j = \mathbf{m}^T \mathbf{x}_{j-m+1:j} \,. \tag{1}$$

Usually, $\mathbf{x}_i$ is not a single value, but a d-dimensional word vector so that $\mathbf{x} \in \mathbb{R}^{d \times n}$. There exist two types of 1d convolution operations. One was introduced by (Waibel et al., 1989) and also known as Time Delay Neural Networks (TDNNs). The other one was introduced by (Collobert et al., 2011). In TDNN, weights $\mathbf{m} \in \mathbb{R}^{d \times m}$ form a matrix. Each row of $\mathbf{m}$ is convolved with the corresponding row of $\mathbf{x}$. In (Collobert et al., 2011) architecture, a sequence of length n is represented as:

$$\mathbf{x}_{1:n} = \mathbf{x}_1 \oplus \mathbf{x}_2 \cdots \oplus \mathbf{x}_n \,, \tag{2}$$

where $\oplus$ is the concatenation operation. In general, let $\mathbf{x}_{i:i+j}$ refer to the concatenation of words $\mathbf{x}_i, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_{i+j}$. A convolution operation involves a filter $\mathbf{w} \in \mathbb{R}^{hk}$, which is applied to a window of h words to produce the new feature. For example, a feature c_i is generated from a window of words $\mathbf{x}_{i:i+h-1}$ by:

$$c_i = f(\mathbf{w} \cdot \mathbf{x}_{i:i+h-1} + b) \,, \tag{3}$$

where $b \in \mathbb{R}$ is a bias term and f is a non-linear function such as the hyperbolic tangent. This filter is applied to each possible window of words in the sequence $\{\mathbf{x}_{1:h}, \mathbf{x}_{2:h+1}, \ldots, \mathbf{x}_{n-h+1:n}\}$ to produce the feature map:

$$\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \,, \tag{4}$$

where $\mathbf{c} \in \mathbb{R}^{n-h+1}$.

We also employ dropout on the penultimate layer with a constraint on ℓ_2-norms of the weight vector. Dropout prevents co-adaptation of hidden units by randomly dropping out a proportion p of the hidden units during forward-backpropagation. That is, given the penultimate layer $\mathbf{z} = [\hat{c_1}, \ldots, \hat{c_m}]$, instead of using:

$$y = \mathbf{w} \cdot \mathbf{z} + b \tag{5}$$

for output unit y in forward propagation, dropout uses:

$$y = \mathbf{w} \cdot (\mathbf{z} \circ \mathbf{r}) + b \,, \tag{6}$$

where $\circ$ is the element-wise multiplication operator and $\mathbf{r} \in \mathbb{R}^m$ is a masking vector of Bernoulli random variables with probability p of being 1. Gradients are backpropagated only through the unmasked units. At test step, the learned weight vectors are scaled by p such that $\hat{\mathbf{w}} = p\mathbf{w}$, and $\hat{\mathbf{w}}$ is used to score unseen sentences. We additionally constrain l_2-norms of the weight vectors by re-scaling $\mathbf{w}$ to have $||\mathbf{w}||_2 = s$ whenever $||\mathbf{w}||_2 > s$ after a gradient descent step.

4 Experimental Results

4.1 Dataset

We use the Clinical TempEval corpus [3] as the evaluation dataset. This corpus was based on a set of

[3] http://alt.qcri.org/semeval2016/task12/index.php?id=data

Category	Train	Dev	Test
Documents	293	147	151
Events	38872	20973	18989

Table 2: Number of documents, event expressions in the training, development and testing portions of the THYME data

600 clinical notes and pathology reports from cancer patients at the Mayo Clinic. These notes were manually de-identified by the Mayo Clinic to replace names, locations, *etc*. with generic placeholders, but time expression were not altered. The notes were then manually annotated with times, events, and temporal relations in clinical notes. These annotations include time expression types, event attributes, and an increased focus on temporal relations. The event, time, and temporal relation annotations were distributed separately from the text using the Anafora standoff format. Table 2 shows the number of documents, event expressions in the training, development and testing portions of the 2016 THYME data.

4.2 Evaluation Metrics

All of the tasks were evaluated using the standard metrics of precision (P), recall (R), and F_1:

$$\begin{aligned} P &= \frac{|S \cap H|}{|S|} \\ R &= \frac{|S \cap H|}{|H|} \\ F_1 &= \frac{2 \cdot P \cdot R}{P + R} \end{aligned} \tag{7}$$

where S is the set of items predicted by the system and H is the set of items manually annotated by the humans. Applying these metrics of the tasks only requires a definition of what is considered an "item" for each task. For evaluating the spans of event expressions, items were tuples of character offsets. Thus, system only received credit for identifying events with exactly the same character offsets as the manually annotated ones. For evaluating the attributes of event expression types, items were tuples of (begin, end, value) where begin and end are character offsets and value is the value that was given to the relevant attribute. Thus, systems only received credit for an event attribute if they both found an event with correct character offsets and then assigned the correct value for that attribute (Bethard et al., 2015).

4.3 Hyperparameters and Training Details

Objective Function

We want to maximize the likelihood of the correct class. This is equivalent to minimizing the negative log-likelihood (NLL). More specifically, the label $\hat{y}$ given the inputs x_h is predicted by a softmax classifier that takes the hidden state h_j as input:

$$\begin{aligned} \hat{p}_\theta(y|x_h) &= softmax(W \cdot x_h + b) \\ \hat{y} &= \underset{y}{\operatorname{argmax}}\, \hat{p}_\theta(y|x_h) \end{aligned} \tag{8}$$

After that, the objective function is the negative log-likelihood of the true class labels y^k:

$$J(\theta) = -\frac{1}{m} \sum_{k=1}^{m} \log \hat{p}_\theta(y^k|x_h^k) + \frac{\lambda}{2}||\theta||_2^2\,, \tag{9}$$

where m is the number of training examples and the superscript k indicates the k-th example.

Hyperparameters

We use Lasagne [4] deep learning framework. We first initialize our word representations using publicly available 300-dimensional Glove word vectors [5]. We deploy CNN model with kernel width of 2, a filter size of 300, sequence length is $2 * windows\_size+1$, number filters is $seqlen-kw+1$, stride is 1, pool size is $seqlen - filter\_size + 1$, cnn activation function is tangent, MLP activation function is sigmoid. MLP hidden dimension is 50. We initialize CNN weights using a uniform distribution. Finally, by stacking a softmax function on top, we can get normalized log-probabilities. Training is done through stochastic gradient descent over shuffled mini-batches with the AdaGrad update rule (Duchi et al., 2011). The learning rate is set to 0.05. The mini-batch size is 100. The model parameters were regularized with a per-minibatch L2 regularization strength of 10^{-4}.

[4] https://github.com/Lasagne/Lasagne

[5] http://nlp.stanford.edu/projects/glove/

4.4 Results and Discussions

Table 3 shows results on the event expression tasks. Our initial submits RUN 4 and 5 outperformed the memorization baseline on every metric on every task. The precision of event span identification is close to the max report. However, our system got lower recall. The reason of lower recall values is that the word2vec may not cover more domain specific words. Table 4 shows results on the phase 2 subtask.

	DocTimeRel		
Methods	P	R	F1
Memorize	-	0.675	-
Ours RUN5	0.788	0.788	0.788
Ours RUN6	0.786	0.786	0.786
Median report	-	0.724	-
Max report	-	0.843	-

Table 4: Phase 2: DocTimeRel

5 Conclusions

In this paper, we introduced a new clinical information extraction system that only leverages deep neural networks to identify event spans and their attributes from raw clinical notes. We trained deep neural networks based classifiers to extract clinical event spans. Our method attached each word to their part-of-speech tag and shape information as extra features. We then hire temporal convolution neural network to learn hidden feature representations. The entire experimental results demonstrate that our approach consistently outperforms the existing baseline methods on standard evaluation datasets.

Our research proved that we can get competitive results without the help of a domain specific feature extraction toolkit, such as cTAKES. Also we only leverage basic natural language processing modules such as tokenization and part-of-speech tagging. With the help of deep representation learning, we can dramatically reduce the cost of clinical information extraction system development. Due to the recall values are still low, we will consider use domain specific tools to enhance feature engineering.

References

Steven Bethard, Leon Derczynski, Guergana Savova, Guergana Savova, James Pustejovsky, and Marc Verhagen. 2015. Semeval-2015 task 6: Clinical tempeval. *Proc. SemEval*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

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

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 2014. A neural network for factoid question answering over paragraphs. In *Empirical Methods in Natural Language Processing*.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

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.

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, pages 801–809.

Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Sumithra Velupillai, Danielle L Mowery, Samir Abdelrahman, Lee Christensen, and Wendy W Chapman. 2015. Blulab: Temporal information extraction for the 2015 clinical tempeval challenge. Association for Computational Linguistics.

Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. 1989. Phoneme recognition using time-delay neural networks. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(3):328–339.

Methods	span			modality			degree			polarity			type		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Memorize	0.878	0.834	0.855	0.810	0.770	0.789	0.874	0.831	0.852	0.812	0.772	0.792	0.855	0.813	0.833
Ours RUN4	0.908	0.842	0.874	0.842	0.780	0.810	0.904	0.838	0.869	0.876	0.812	0.842	0.877	0.813	0.844
Ours RUN5	0.900	0.850	0.874	0.837	0.790	0.813	0.896	0.845	0.870	0.861	0.813	0.836	0.869	0.820	0.844
Median report	0.887	0.846	0.874	0.830	0.780	0.810	0.882	0.838	0.869	0.868	0.813	0.839	0.854	0.813	0.844
Max report	0.915	0.891	0.903	0.866	0.843	0.855	0.911	0.887	0.899	0.900	0.875	0.887	0.894	0.870	0.882

Table 3: System performance comparisons. Note that Run4 means the window size is 4, and Run5 means the window size is 5

Wenpeng Yin and Hinrich Schutze. 2015. Multigrancnn: An architecture for general matching of text chunks on multiple levels of granularity. In *Proceedings of th 53rd Annual Meeting of the Association for Computational Linguistics*, pages 63–73.