# TCS Research at TAC 2017: Joint Extraction of Entities and Relations from Drug Labels using an Ensemble of Neural Networks

**Sachin Pawar**[1,2], **Girish K. Palshikar**[1], **Pushpak Bhattacharyya**[2], **Nitin Ramrakhiyani**[1,3], **Shashank Gupta**[3], and **Vasudeva Varma**[3]

[1]TCS Research, Tata Consultancy Services, Pune
[2]Indian Institute of Technology Bombay, Mumbai
[3]International Institute of Information Technology, Hyderabad
{sachin7.p,gk.palshikar}@tcs.com
pb@cse.iitb.ac.in, nitin.ramrakhiyani@tcs.com
shashank.gupta@research.iiit.ac.in,vv@iiit.ac.in

## Abstract

We describe our submission at TAC 2017 for extracting entities and relations of interest from drug labels. We employ an end-to-end relation extraction system which jointly extracts both entities and relations. The task of end-to-end relation extraction consists of identifying boundaries of entity mentions, entity types of these mentions and appropriate semantic relation for each pair of mentions. Based on our earlier work (Pawar et al., 2017), a single neural network model ("All Word Pairs" model i.e. AWP-NN) is trained to assign an appropriate label to each word pair in a given sentence for performing end-to-end relation extraction. Moreover, we build an ensemble of multiple AWP-NN models to achieve better performance than the individual models. We achieved 73.18% and 24.79% F-measures for entity and relation extraction, respectively.

## 1 Introduction

Our team TRDDC_IIITH participated in two tasks of the TAC 2017 track "Adverse Reaction Extraction from Drug Labels" : Task 1 (extracting entities of interest) and Task 2 (identifying relations of interest among the extracted entities). Task 1 is similar to traditional NLP task of Named Entity Recognition and Task 2 is similar to relation identification task. In this paper, we describe our system for TAC 2017 submission. Here, entity types of interest in Task 1 are : AdverseReaction, Severity, Factor, DrugClass, Negation and Animal. Relations of interest in Task 2 are : Negated, Hypothetical and Effect.

The task of end-to-end relation extraction consists of three sub-tasks: i) identifying boundaries of entity mentions, ii) identifying entity types of these mentions and iii) identifying appropriate semantic relation for each pair of mentions. This is in contrast with pure relation extraction techniques (GuoDong et al., 2005; Jiang and Zhai, 2007; Bunescu and Mooney, 2005; Qian et al., 2008) which assume that for a sentence, gold-standard entity mentions (i.e. boundaries as well as types) in it are known. We propose to use end-to-end relation extraction system to jointly address both the tasks of entity extraction (Task 1) and relation extraction (Task 2).

Traditionally, the three sub-tasks of end-to-end relation extraction are carried out serially in a "pipeline" fashion. In this case, the errors in any sub-task affect subsequent sub-tasks. Another disadvantage of this "pipeline" approach is that it allows only one-way *information flow*, i.e. the knowledge about entities is used for identifying relations but not vice versa. Hence to overcome this problem, several approaches (Roth and Yih, 2004; Roth and Yih, 2002; Singh et al., 2013; Li and Ji, 2014) were proposed which carried out these sub-tasks jointly rather than in "pipeline" manner.

In our earlier work (Pawar et al., 2017), we proposed the "All Word Pairs" neural network model (AWP-NN) which reduces solution of the three sub-tasks to predicting an appropriate label for each word pair in a given sentence. End-to-end relation extraction output can then be constructed easily from these labels of word pairs. In this work, we propose to create an ensemble of multiple such AWP-NN models to achieve better performance.

## 2 Problem Definition

Given a sentence as an input, an end-to-end relation extraction system is expected to produce a list of entity mentions within it. For each entity men-

| Entity Mention | Entity Type |
|---|---|
| allergic$_5$ reactions$_6$ | AdverseReaction |
| anaphylaxis$_8$ | AdverseReaction |
| may$_9$ | Factor |

Table 1: Expected output of end-to-end relation extraction system for entity mentions

| Entity Mention Pair | Relation Type |
|---|---|
| allergic reactions, may | Hypothetical |
| anaphylaxis, may | Hypothetical |

Table 2: Expected output of end-to-end relation extraction system for relations

tion, its boundaries and entity type should be identified. Also, for each pair of valid entity mentions, it should decide whether any pre-defined semantic relation holds between them.

Consider the sentence : Like$_0$ any$_1$ injectable$_2$ drug$_3$ ,$_4$ allergic$_5$ reactions$_6$ and$_7$ anaphylaxis$_8$ may$_9$ occur$_{10}$ .$_{11}$ Here, end-to-end relation extraction should produce the output as shown in the tables 1 and 2.

## 3 All Word Pairs Model (AWP-NN)

We employ the AWP-NN (All Word Pairs model using Neural Networks) model (Pawar et al., 2017) which is a single joint model for addressing all three sub-tasks of end-to-end relation extraction : i) identifying boundaries of entity mentions, ii) identifying entity types of these mentions and iii) identifying appropriate semantic relation for each pair of mentions. Here, annotations of all these three sub-tasks are represented by assigning an appropriate label to each pair of words. It is not necessary to assign label to all possible word pairs; rather $i^{th}$ word is paired with $j^{th}$ word only when $j \geq i$. AWP-NN model is motivated from the table representation idea proposed by Miwa and Sasaki (2014) but differs significantly from it in following ways:

1. boundary identification is modelled with the help of a special relation type (WEM) instead of BIO (**B**egin, **I**nside, **O**ther) encoding or BILOU (**B**egin, **I**nside, **L**ast, **U**nit, **O**ther) encoding

2. neural network model for prediction of appropriate label for each word pair instead of structured prediction

Labels predicted by the AWP-NN model for each word pair are then used for constructing the end-to-end relation extraction output as described in tables 1 and 2.

Consider the example sentence from Section 2. Table 3 shows true annotations of all word pairs in this sentence as required for training the AWP-NN model. Overall, 12 labels are used for this annotation, which can be grouped into the following 5 logical clusters:

1. AdverseReaction, Severity, Factor, DrugClass, Negation and Animal : Represent entity type of *head word* of an entity mention when both the words in a word pair are the same

2. OTH : Represents words which are not head words of any entity mention and both the words in a word pair are the same

3. Negated, Hypothetical and Effect : Represent relation type between *head words* of any two entity mentions

4. NULL : Indicates that no pre-defined semantic relation exists between the words in the word pair

5. WEM (Within Entity Mention) : Indicates that the words in the word pair belong to the same entity mention and one of the word is the *head word* of that mention

**Head word of an entity mention:** In order to annotate word pairs with an appropriate label as described above, it is important to identify head words of entity mentions. For single word entity mentions, identifying head word is trivial. For any multi-word entity mention, we examine each word in that mention from right hand side. If dependency parent of any word lies outside the entity mention or it is the root of the sentence, then that word is identified as the head word. Following are some examples of mentions and their respective head words:

1. anaphylaxis $\leftrightarrow$ anaphylaxis

2. allergic reactions $\leftrightarrow$ reactions

3. pain in stomach $\leftrightarrow$ pain

| | Like | any | injectable | drug | , | allergic | reactions | and | anaphylaxis | may | occur | . |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Like | O | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ |
| any | | O | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ |
| injectable | | | O | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ |
| drug | | | | O | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ |
| , | | | | | O | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ |
| allergic | | | | | | O | WEM | ⊥ | ⊥ | ⊥ | ⊥ | ⊥ |
| reactions | | | | | | | AR | ⊥ | ⊥ | Hypo | ⊥ | ⊥ |
| and | | | | | | | | O | ⊥ | ⊥ | ⊥ | ⊥ |
| anaphylaxis | | | | | | | | | AR | Hypo | ⊥ | ⊥ |
| may | | | | | | | | | | Factor | ⊥ | ⊥ |
| occur | | | | | | | | | | | O | ⊥ |
| . | | | | | | | | | | | | O |

Table 3: Annotation of all word pairs as per the AWP-NN model (AR: AdverseReaction, Hypo: Hypothetical, O: OTH, ⊥: NULL)

### 3.1 Features for the AWP-NN model

Each word pair constitutes a separate instance for classification for the AWP-NN model. Features used by AWP-NN model are of three types: i) features characterizing individual word in a word pair, ii) features characterizing properties of both the words at a time and iii) features based on feedback, i.e. predictions of preceding instances.

#### 3.1.1 Individual word features

These features are generated separately for both the words in a word pair.
1. Word itself and its POS tag
2. Previous word and previous POS tag
3. Next word and next POS tag
4. Parent / Governor of the word in the dependency tree, the corresponding dependency relation type and POS tag of the parent

#### 3.1.2 Word pair features

These features are generated for a word pair (say $\langle W_i, W_j \rangle$) as a whole.
1. Words distance (WD): Number of words in the sentence between the words $W_i$ and $W_j$
2. Tree distance (TD): Number of words on the path leading from $W_i$ to $W_j$ in the sentence's dependency tree
3. Common Ancestor (CA): Lowest common ancestor of the two words in the dependency tree
4. Ancestor Position (AP): It indicates the position of the common ancestor with respect to the two words of a word pair. Different possible positions of the ancestor are - left of $W_i$, $W_i$ itself, between $W_i$ and $W_j$, $W_j$ itself and right of $W_j$.
5. Dependency Path ($DP_1, DP_2, \cdots, DP_K$) : Sequence of dependency relation types (ignoring directions) on the dependency path leading from $W_i$ to $W_j$ in the sentence's dependency tree.

#### 3.1.3 Feedback features

These features are based on predictions of the preceding instances. Unlike other sequence labelling problems such as Named Entity Recognition where each word gets a label and there is natural order / sequence of instances (i.e. words), there is no natural order / sequence of instances (i.e. word pairs) for AWP-NN model. Hence, for each instance we identify its two preceding instances and define two corresponding feedback features ($FB_1$ and $FB_2$). Let $\langle W_i, W_j \rangle$ be an instance representing a word pair in a sentence having $N$ words such that $1 \leq i, j \leq N$ and $i \leq j$. There are following two cases for identifying two preceding instances of $\langle W_i, W_j \rangle$:

- If $i = j$ then both the preceding instances are same i.e. $\langle W_{i-1}, W_{i-1} \rangle$. Feedback features: $FB_1 = FB_2 = LabelOf(\langle W_{i-1}, W_{i-1} \rangle)$
- If $i < j$ then the preceding instances are $\langle W_i, W_i \rangle$ and $\langle W_j, W_j \rangle$. Feedback features: $FB_1 = LabelOf(\langle W_i, W_i \rangle)$ and $FB_2 = LabelOf(\langle W_j, W_j \rangle)$

Label predictions of the preceding instances are then represented using one-hot encoding and used as features. During training, true labels of the preceding instances are used but while decoding, the predicted labels of these instances are used. Hence during decoding, predictions for word pairs of the form $\langle W_i, W_i \rangle$ (diagonal word pairs in the table 3) are obtained first, starting from $i = 1$ to $N$. Predictions of other word pairs can be obtained later, as predictions of their preceding instances would then be available.

### 3.2 Architecture of the AWP-NN model

Figure 1 shows various major components in the architecture of the AWP-NN model.

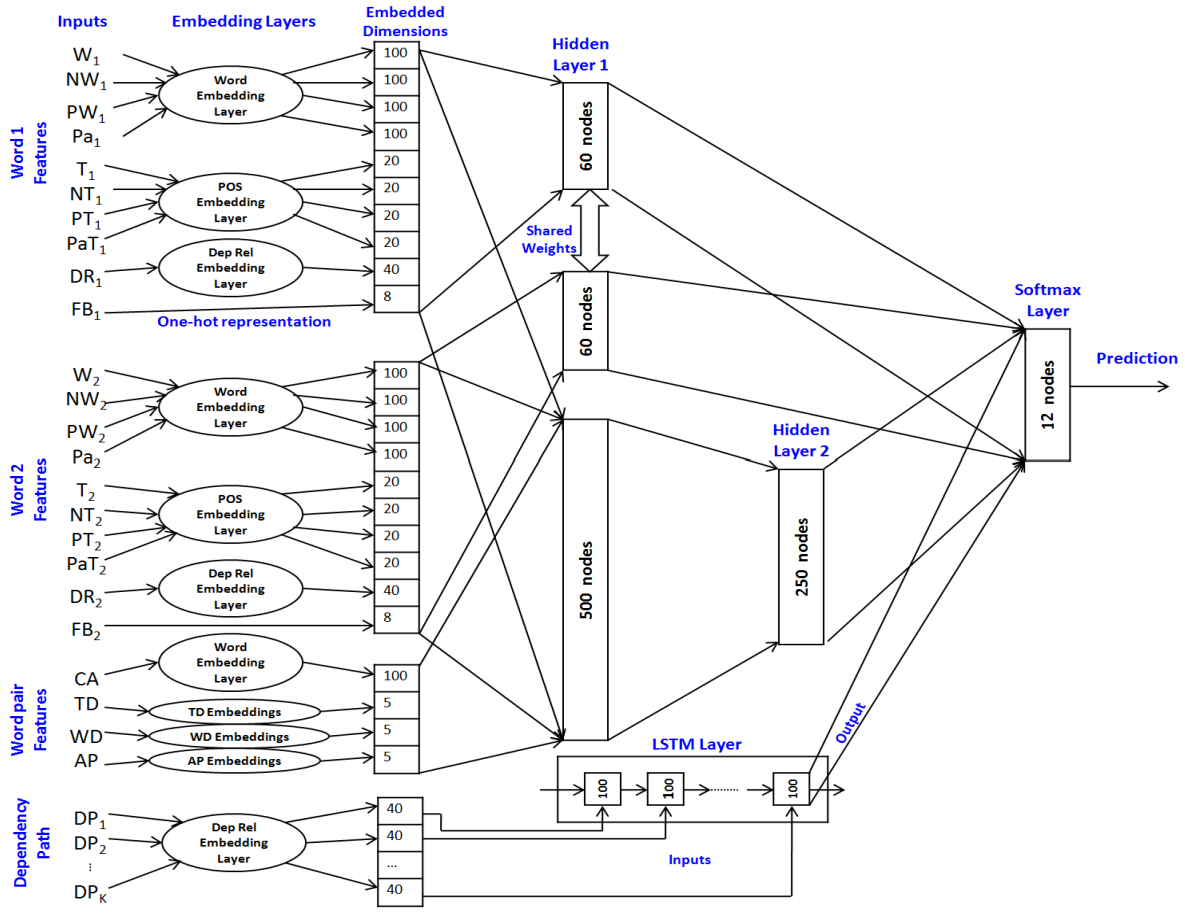


Figure 1: AWP-NN model architecture for predicting appropriate label for the given word pair. ($W_1, W_2$: words in the word pair; $NW_1, PW_1, NW_2, PW_2, NT_1, PT_1, NT_2, PT_2$: next and previous words/POS tags of $W_1$ and $W_2$; $Pa_1, DR_1, Pa_2, DR_2$: parents and corresponding dependency relation types of $W_1$ and $W_2$ in the dependency tree; $PaT_1, PaT_2$: POS tags of the parents of $W_1$ and $W_2$; $FB_1, FB_2$: Predictions of the preceding instances; $CA$: Lowest common ancestor of $W_1$ and $W_2$ in the dependency tree; $TD$: Tree distance; $WD$: Words distance; $AP$: Ancestor position; $DP_1, DP_2, \cdots, DP_K$: Sequence of dependency relation types on the dependency path leading from $W_1$ to $W_2$; Embedding layers for words, POS and dependency relations are shown separately for clarity, but are shared throughout the network.

### 3.2.1 Embedding Layers

Most of the features used by the model are discrete in nature such as words, POS tags, dependency relation types and ancestor position. These discrete features have to be mapped to some numerical representation and *embedding* layers are used for this purpose. We have employed following embedding layers to represent various types of features:

**Word embedding layer**: It maps each word to a real-valued vector of some fixed dimensions. We initialize this layer with the *pre-trained* 100 dimensional GloVe word vectors[1] learned on Wikipedia corpus. All the different features which are expressed in the form of words ($W_1$, $W_2$, $NW_1$, $PW_1$, $NW_2$, $PW_2$, $Pa_1$, $Pa_2$ and $CA$ in the figure 1) share the *same* word embedding layer. During training, the initial embeddings get *fine-tuned* for our supervised classification task.

**POS embedding layer**: It maps each distinct POS tag to some real-valued vector representation. All the different features which are expressed in the form of POS tags ($T_1$, $T_2$, $NT_1$, $PT_1$, $NT_2$, $PT_2$, $PaT_1$ and $PaT_2$ in the figure 1) share the *same* embedding layer.

**Dependency relation type embedding layer**: It maps each distinct dependency relation type to some real-valued vector representation. All the features based on dependency types ($DR_1, DR_2, DP_1, \cdots, DP_K$ in the figure 1) also share the *same* embedding layer.

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

**AP embedding layer**: It maps each distinct ancestor position to some real-valued vector representation.

**WD/TD embedding layer**: Even though word distance (WD) and tree distance (TD) are numerical features, we used embeddings to represent each distinct value for them as range of values of these features is large. It was observed to be better than directly providing them as inputs to the neural network.

In our experiments, we used 20 dimensions for POS embeddings, 40 for dependency relation type embeddings and 5 dimensions for AP, WD and TD embeddings. Unlike word embeddings these were initialized randomly during training.

#### 3.2.2 Hidden Layers

First hidden layer is divided in 3 parts. First two parts of 60 nodes each are connected to only the features capturing first and second word, respectively. These nodes are expected to capture higher level abstract features of both the words separately. In order to force these two parts to learn similar abstract features, the weights matrix is shared among them. The third part of the first hidden layer consisting of 500 nodes is connected to all the input features except dependency path, i.e. individual word features of two words, word pair features and feedback features. Output of this part is further given as input to the second hidden layer of 250 units. Output of the second hidden layer is fed to the final softmax layer. Also, outputs of the first two parts of the first hidden layer are directly connected to the final softmax layer. As the dependency path is represented as a sequence of dependency relation types, it is fed to a separate LSTM layer. Output of the LSTM layer is directly connected to the final softmax layer. Softmax layer consists of 12 nodes, each representing one of the possible prediction label described earlier.

### 3.3 Ensemble of multiple AWP-NN models

As neural networks are initialized randomly, if we train a neural network model multiple times, each time it converges to a different set of model parameters and hence different predictions are obtained each time. Hence, we propose to train multiple such models and create an ensemble classifier with multiple AWP-NN models as base classifiers.

AWP-NN model predicts one of the class label out of 12 possible classes, for each word pair. It also outputs a probability distribution over these 12 classes. As discussed already, each different run produces a different set of AWP-NN model parameters and hence produces different probability distributions for each word pair. We train $n$ different AWP-NN models as a result of $n$ different runs. An ensemble of $n$ AWP-NN models is created by simply averaging out their $n$ output probability distributions for each word pair. It can be observed in the table 4 that the ensemble models perform much better than stand-alone models.

### 3.4 Filtering Invalid Mentions and Relations

Task 1 specified that other than AdverseReactions, mentions are only annotated when they are related to an AdverseReaction by one of the relations (Negated, Hypothetical and Effect). Hence, we consider mentions of entity types other than AdverseReaction as invalid if they are not related to any mention of AdverseReaction. Such invalid mentions are removed as a post-processing step.

For relation type in Task 2, valid entity types of the argument mentions are specified. The relation type Negated relates AdverseReaction to either Negation or Factor. The relation type Hypothetical relates AdverseReaction to either Animal, DrugClass or Factor. Similarly, the relation type Effect relates AdverseReaction to its Severity. If any relation is predicted between two entity mentions such that their entity types are not compatible with the predicted relation type, then it is considered as invalid prediction. And such invalid relation predictions are removed as a post-processing step.

## 4 Experimental Analysis

### 4.1 Dataset

We used 100 annotated drug labels for training our AWP-NN models. The data was pre-processed using Stanford CoreNLP toolkit (Manning et al., 2014) for sentence detection, tokenization, Part-of-speech tagging and dependency parsing.

For tuning the parameters, we did not use TAC 2017 drug labels dataset. Rather we used the best parameter settings observed for the ACE 2004 dataset (Doddington et al., 2004) in our previous work (Pawar et al., 2017). ACE 2004 dataset is the most widely used dataset for reporting relation extraction performance. The details of entity and relation types can be found in Pawar et al. (2017).

### 4.2 Implementation details

We used Keras (Chollet, 2015) for implementing our AWP-NN model. The model was trained for 40 epochs using batch size of 64 instances. We used *Dropout* (Srivastava et al., 2014) for regularization with probability 0.5 for hidden layers and 0.1 for embedding layers. The value of $K$ (maximum length of dependency path, see Figure 1) was set to be 4, hence all word pairs having length of dependency path more than 4 were assumed to have NULL label.

### 4.3 Results

The results in Tables 4 and 5 are divided in two different sections:
1. **only entity extraction:** It includes boundary identification as well as entity type classification.
2. **entity+relation extraction:** It is end-to-end relation extraction which includes boundary identification, entity type classification and relation type classification. Here, correct relation label for an entity mention pair is counted as a true positive only if boundaries and entity types of both the mentions are identified correctly.

Table 4 shows the comparative performances (in terms of micro-F1 measure) for various approaches on the ACE 2004 dataset. It can be observed that our ensemble approach improves the end-to-end relation extraction performance significantly by 3% F-measure. Hence, we employ the same approach with the same parameter settings for entity and relation extraction from TAC 2017 drug labels dataset. Table 5 shows the entity and relation extraction performance on the drug labels dataset. The first row shows the results as per the submitted version of our system output. 5 base classifiers (AWP-NN models) were used in the ensemble for ACE 2004 dataset whereas 3 base classifiers were used for drug labels dataset. After submission, we kept on analyzing the errors and made some minor changes to word pair annotations in training data. The second row shows the revised results where relation extraction performance has improved.

## 5 Conclusion and Future Work

In this paper, we described our submission at TAC 2017 for extracting entities and relations of interest from drug labels. We employed an end-to-end relation extraction system which jointly extracts both entities and relations. We used AWP-NN (All Word Pairs model using Neural Networks) based on our earlier work (Pawar et al., 2017). AWP-NN model is trained to assign an appropriate label to each word pair in a given sentence for performing end-to-end relation extraction. Moreover, we built an ensemble of multiple AWP-NN models to achieve better performance than the individual models. We achieved 73.18% and 24.79% F-measures for entity and relation extraction, respectively.

In this work, we have not used any specific domain knowledge from the health domain. In future, we plan to explore various ways to incorporate such domain knowledge. Instead of using word embeddings pre-trained on the Wikipedia corpus, we would like to experiment with health domain specific word embeddings. Also, we will explore the use of Markov Logic Networks as they are used in (Pawar et al., 2017) for incorporating domain knowledge in the form of weighted first order logic rules.

## References


Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731. ACL.

Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 551–560. ACL.

Franois Chollet. 2015. keras. `https://github.com/fchollet/keras`.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel. 2004. The Automatic Content Extraction (ACE) Program-Tasks, Data, and Evaluation. In *LREC*, volume 2, page 1.

Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 427–434. Association for Computational Linguistics.

Jing Jiang and ChengXiang Zhai. 2007. A Systematic Exploration of the Feature Space for Relation Extraction. In *HLT-NAACL*, pages 113–120.

| Approach | Entity Extraction | | | Entity+Relation | | |
|---|---|---|---|---|---|---|
| | P | R | F | P | R | F |
| AWP-NN (Pawar et al., 2017) | 81.1 | 79.7 | 80.4 | 55.6 | 44.4 | 49.3 |
| AWP-NN Ensemble | 82.7 | 80.7 | 81.7 | 62.1 | 45.1 | **52.3** |

Table 4: Comparative performance of an ensemble of AWP-NN models with average of individual AWP-NN models on the ACE 2004 dataset. The numbers are micro-averaged and obtained after 5-fold cross-validation.

| Approach | Entity Extraction (Task 1) | | | Entity+Relation (Task 2) | | |
|---|---|---|---|---|---|---|
| | P | R | F | P | R | F |
| AWP-NN Ensemble (Submitted) | 76.07 | 70.49 | 73.18 | 37.43 | 18.53 | 24.79 |
| AWP-NN Ensemble (Revised) | 76.85 | 69.54 | 73.01 | 45.77 | 22.78 | 30.42 |

Table 5: Performance of an ensemble of AWP-NN models on the TAC 2017 drug labels dataset.


Rohit J Kate and Raymond J Mooney. 2010. Joint entity and relation extraction using card-pyramid parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 203–212. ACL.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *ACL*.

Qi Li, Heng Ji, Yu Hong, and Sujian Li. 2014. Constructing information networks using one single model. In *EMNLP*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*.

Makoto Miwa and Yutaka Sasaki. 2014. Modeling Joint Entity and Relation Extraction with Table Representation. In *EMNLP*, pages 1858–1869.

Sachin Pawar, Pushpak Bhattacharyya, and Girish Palshikar. 2016. End-to-end relation extraction using markov logic networks. In *Proceedings of the 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2016), LNCS 9624*. Springer.

Sachin Pawar, Pushpak Bhattacharyya, and Girish Palshikar. 2017. End-to-end Relation Extraction using Neural Networks and Markov Logic Networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 818–827.

Longhua Qian, Guodong Zhou, Fang Kong, Qiaoming Zhu, and Peide Qian. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 697–704. ACL.

Dan Roth and Wen-tau Yih. 2002. Probabilistic reasoning for entity & relation recognition. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. ACL.

D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *CoNLL*, pages 1–8.

Dan Roth and Wen-tau Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. *Introduction to statistical relational learning*, pages 553–580.

Sameer Singh, Sebastian Riedel, Brian Martin, Jiaping Zheng, and Andrew McCallum. 2013. Joint inference of entities, relations, and coreference. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 1–6. ACM.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.