Joint Models for Extracting Adverse Drug Events from Biomedical Text

Fei Li,[1] **Yue Zhang,**[2] **Meishan Zhang,**[3] **Donghong Ji**[1*]
1. State Key Laboratory of Software Engineering, School of Computer, Wuhan University, China
2. Singapore University of Technology and Design
3. School of Computer Science and Technology, Heilongjiang University, China
{lifei_csnlp, dhji}@whu.edu.cn
yue_zhang@sutd.edu.sg, mason.zms@gmail.com

Abstract

Extracting adverse drug events receives much research attention in the biomedical community. Previous work adopts pipeline models, firstly recognizing drug/disease entity mentions and then identifying adverse drug events from drug/disease pairs. In this paper, we investigate joint models for simultaneously extracting drugs, diseases and adverse drug events. Compared with pipeline models, joint models have two main advantages. First, they make use of information integration to facilitate performance improvement; second, they reduce error propagation in pipeline methods. We compare a discrete model and a deep neural model for extracting drugs, diseases and adverse drug events jointly. Experimental results on a standard ADE corpus show that the discrete joint model outperforms a state-of-the-art baseline pipeline significantly. In addition, when discrete features are replaced by neural features, the recall is further improved.

1 Introduction

Automatically extracting adverse drug events (ADEs) has recently received much research attention in the biomedical community [Gurulingappa *et al.*, 2012; Yildirim *et al.*, 2013; Kang *et al.*, 2014; Yildirim *et al.*, 2014; Liu *et al.*, 2014; Yates *et al.*, 2015; Wei *et al.*, 2015]. As shown in Figure 1, the task is to identify mentions of drugs and their side effects, such as diseases that they cause, from a piece of raw text. It plays important roles in pharmacovigilance and biocuration.

Traditionally, the extraction of ADEs involves two steps. First, mentions of drug/disease entities are recognized in a given sentence. This subtask is similar in nature to the task of named entity recognition (NER) [Li and Ji, 2014]. Second, each drug/disease pair is examined to decide whether they have an ADE relation. This step can be casted as a classification problem and solved using statistical models such as Support Vector Machines (SVMs) [Zhou *et al.*, 2005].

The method above involves a two-step pipeline, where submodels of the steps are trained separately. One major disad-

Gliclazide$_{drug}$-induced **acute hepatitis**$_{disease}$.
Hepatitis$_{disease}$ caused by **methotrexate**$_{drug}$ and **etretinate**$_{drug}$.
Vitiligo$_{disease}$ associated with **alpha interferon**$_{drug}$ in a patient with chronic active hepatitis C.
A woman who was treated for thyrotoxicosis with **methimazole**$_{drug}$ developed **agranulocytosis**$_{disease}$.

Figure 1: Example ADEs, where drug and disease mentions are shown in green and red, respectively.

vantage in this method is error propagation: if a drug or disease is incorrectly recognized, the recognition of its related ADEs will be incorrect. Joint models, which process all subtasks simultaneously, have been proposed to avoid error propagation in similar situations, such as word segmentation and part-of-speech (POS) tagging [Zhang and Clark, 2008], NER and parsing [Finkel and Manning, 2009], POS tagging and parsing [Bohnet and Nivre, 2012; Zhang *et al.*, 2012], morphological generation and syntactic linearization [Song *et al.*, 2014], and entity and relation extraction [Roth and Yih, 2007; Chan and Roth, 2011; Li and Ji, 2014]. Such methods exploit the correlations between the relevant subtasks for mutual benefit. We investigate joint entity recognition and ADE extraction by solving the two subtasks using a single model. One additional advantage of joint methods is that they can model the interactions between the drug/disease and ADE information, which cannot be handled by pipeline methods.

The combined search space of a joint task can be significantly larger than those of its subtasks, thereby making search a challenging problem. Exact inference for our joint task can be highly inefficient. To this end, we propose a transition-based model to extract drugs, diseases and ADEs jointly. Such methods [Nivre, 2008; Zhang and Clark, 2011] cast the output-building process as a state-transition process, where states correspond to partial outputs and transition actions build outputs incrementally. Statistical models are built to score transition actions using non-local features, and greedy or beam-search is typically used to address the search challenge. The method is particularly suitable for joint tasks with complex search space.

We investigate both discrete models and deep neural networks for our joint task. Neural models have been explored

[*]Corresponding author

in a range of tasks of natural language processing (NLP) [Collobert *et al.*, 2011; Cho *et al.*, 2014; Chen and Manning, 2014]. Compared with discrete models using high-dimensional features, neural models adopt low-dimensional dense embedding features, and can effectively settle the feature sparsity problem. Furthermore, non-linear hidden layers of neural models can be used to extract salient features and combine them without any human interventions.

Experimental results on a standard ADE corpus [Gurulingappa *et al.*, 2012] show that the discrete joint model significantly outperforms a strong pipelined baseline. In addition, the neural joint model can further improve the performance over the discrete joint model by significantly boosting the recall. With some small modifications, our joint models can be used for other tasks of entity-relation extraction such as protein-protein interactions.

2 Related Work

Previous work seperates the task of ADE extraction into *drug/disease entity recognition* and *ADE relation extraction*. NER models can be directly applied for the first subtask, and therefore most prior work focuses on the second subtask [Gurulingappa *et al.*, 2012]. A simple way to extract ADEs is based on co-occurrence information of drug/disease pairs [Kandula and Treitler, 2010]. Other approaches use weakly-supervised techniques such as bootstrapping, whose performance depends on seed patterns [Xu and Wang, 2014]. More sophisticated approaches use supervised learning [Liu *et al.*, 2014; Li *et al.*, 2015; Yates *et al.*, 2015], and achieve state-of-the-art performance. We follow this line of work, yet investigating models that jointly identifies entities and relations.

Neural networks have received increasing research attention in the AI community [Collobert *et al.*, 2011; Bengio *et al.*, 2015]. In NLP, neural networks typically take low-dimensional embedding vectors, and use non-linear neural layers for automatic feature combination. It has achieved promising results for NLP tasks such as machine translation [Cho *et al.*, 2014], dependency parsing [Chen and Manning, 2014] and sentiment analysis [Zhang *et al.*, 2015]. Neural models can not only avoid feature sparsity but also learn non-local semantic-level features. To our knowledge, we are the first work to build an information extraction model for jointly extracting entities and relations using neural networks, particularly for drug/disease recognition and ADE extraction.

3 Baseline Pipeline System

The baseline pipeline consists of two submodels, namely a transition-based system for drug/disease recognition and a max-entropy classifier to determine whether a drug/disease pair forms an ADE. Following the state-of-the-art models, discrete features are used in both submodels.

3.1 Drug/Disease Mention Recognition

We use a deterministic transition-based model for drug/disease entity recognition, where the system consists of a set of states and transition actions. Each state corresponds to a partial or full output and actions transform one state to another. For the task of drug/disease recognition, the state transition system is a standard sequence labeling system, where the state consists of a label sequence $\boldsymbol{l}$, and the actions incrementally label the next unlabeled word. We define the actions as:

- O, which marks the current word as not belong to either a drug or disease mention.
- BC, which marks the current word as the beginning of a drug mention.
- BD, which marks the current word as the beginning of a disease mention.
- I, which marks the current word as part of a drug or disease mention but not the beginning.

For example, given a sentence "Gliclazide-induced acute hepatitis.", the action sequence "BC O O BD I O" yields the result "**Gliclazide**$_{drug}$-induced **acute hepatitis**$_{disease}$.".

We use a maximum-entropy classification model for disambiguation. Given a certain state, the model predicts the best next transition action. The feature templates are listed in the top of Table 1, which include word, POS tags, prefix and suffix information at the current location and its context. Semantic features such as WordNet are also utilized.

Following previous work on transition-based NLP [Nivre, 2008; Chen and Manning, 2014], we use greedy search decoding for both the baseline and the joint systems. Results show that the simple and fast search method gives the state-of-the-art accuracies compared with the previous best results.

Features for drug/disease recognition
Word: current word; two words before and after the current word
POS: POS tags of the current word; POS tags of the two words before and after the current word
Prefix and suffix: prefix and suffix of the current word; prefixes and suffixes of the two words before and after the current word
Brown Cluster[1]**:** cluster of the current word; clusters of the two words before and after the current word
WordNet[2]**:** synonym and hypernym of the current word; synonyms and hypernyms of the two words before and after the current word
Ontology: whether the current word is contained by ontology bases such as HumanDO [3], DrugBank [4], CTD [5] and Jochem [6]
Features for ADE extraction
Entity: combinations of the words in both entities involved
Context: two words before each entity mention and two words after each entity mention
WordNet: combinations of synonyms and hypernyms of the entity mentions

Table 1: Feature templates.

[1]https://github.com/percyliang/brown-cluster
[2]https://wordnet.princeton.edu/
[3]https://bioportal.bioontology.org/ontologies/DOID
[4]http://www.drugbank.ca
[5]http://ctdbase.org/
[6]http://biosemantics.org/index.php/resources/jochem

state <***l***, ***ds***, ***dg***, ***s*** >	next action
......	...
<[BD,O,O,BC], [Hepatitis], [], []>	O
<[BD,O,O,BC,O], [Hepatitis], [methotrexate], []>	Y
<[BD,O,O,BC,O,Y], [Hepatitis], [methotrexate], [(Hepatitis,methotrexate)]>	BC
<[BD,O,O,BC,O,Y,BC], [Hepatitis], [methotrexate], [(Hepatitis,methotrexate)]>	O
<[BD,O,O,BC,O,Y,BC,O], [Hepatitis], [methotrexate,etretinate], [(Hepatitis,methotrexate)]>	Y
<[BD,O,O,BC,O,Y,BC,O,Y], [Hepatitis], [methotrexate,etretinate], [(Hepatitis,methotrexate),(Hepatitis,etretinate)] >	<EOS>

Figure 2: State transition examples, where <EOS> denotes the end of sentence.

3.2 ADE extraction

At the second step in the pipeline, ADEs are extracted by traversing each drug/disease pair identified in the previous step, making a binary classification. Because there are no other entity types in our task, and only an drug/disease pair can have an ADE relation, we set the traversing strategy to brute-force enumeration. Only drug/disease pairs need to be enumerated, and drug/drug or disease/disease pairs are ignored.

We use a maximum-entropy classifier for ADE classification. The feature templates are listed in the bottom of Table 1. They consist of words of the entities, their contexts and semantic features such as WordNet.

3.3 Training

Both subtasks of the pipeline baseline employ a maximum-entropy model, and we describe their training in one section for conciseness. Given an input **x**, probability of an output c$_i$ is denoted as:

$$p(c_i|\mathbf{x}) = \frac{e^{\mathbf{w}_i\mathbf{x}}}{\sum_{j=1}^{T} e^{\mathbf{w}_j\mathbf{x}}}, \qquad (1)$$

where **x** denotes the feature vector of the input, p(c$_i$|**x**) denotes the conditional probability of the class c$_i$, $\mathbf{w}_i$ denotes the i-th row of the parameter matrix $\mathbf{W} \in \mathbb{R}^{T\times|\mathbf{x}|}$ and T denotes the number of classes.

We perform maximum-entropy training for both subtasks. For drug/disease recognition, the parameter matrix **W** is tuned to maximize the likelihood of gold-standard actions on each word in the training data. For ADE extraction, the training objective is the likelihood of gold ADE drug/disease pairs. A general form of training objectives is to maximize:

$$L(\mathbf{W}) = -\sum_{j=1}^{|E|} \log p_{g_j} + \frac{\beta}{2}\|\mathbf{W}\|_2^2, \qquad (2)$$

where g$_j$ is the gold class of the j-th example, p$_{g_j}$ is its probability and β is the weight parameter for L$_2$ regularization.

Mini-batched AdaGrad [Duchi *et al.*, 2011] is used for training. For the discrete valued models, our parameter updating method follows Kummerfeld *et al.* [2015].

4 Discrete Joint Model

For information integration and to reduce error propagation, the two submodels of the pipeline are merged into a single discrete joint model, which provides an end-to-end system that identifies both drug/disease mentions and ADEs given a sentence. In consistent with the baseline, a maximum-entropy model with greedy decoding is used for disambiguation.

We leverage a transition-based system for jointly decoding by extending the baseline drug/disease mention recognition system. There are two salient differences: first, the **states** here include not only drug and disease mentions in a sentence, but also ADE relations. Second, in addition to the **actions** used to recognize entities, the transition system also requires **actions** to extract ADE relations. Correspondingly, we define the state of the joint model as a tuple <***l***, ***ds***, ***dg***, ***s*** >, where ***l*** is a label sequence, ***ds*** is a list of readily-recognized disease entity mentions, ***dg*** is a list of readily-recognized drug entity mentions and ***s*** is a set of ADEs. Whenever a new entity has been recognized, it is correlated to each previously identified entity mention to decide whether there is an ADE between the two entities. Two more actions are defined to achieve this.

- N, which indicates that a pair of entities does not have an ADE relation.
- Y, which indicates that a pair of entities has an ADE relation.

Given the sentence "Hepatitis caused by methotrexate and etretinate.", the action sequence "BD O O BC O Y BC O Y" yields the result output "**Hepatitis**$_{disease}$ caused by **methotrexate**$_{drug}$ and **etretinate**$_{drug}$.". Note that the two Y actions are performed after two O actions, which confirms the finishing of two BC entities, respectively. The first Y action is applied when the new drug entity mention "methotrexate" is recognized. At this time, there is only one disease entity "Hepatitis" in the state component ***ds***, and therefore one N/Y action is made. Similarly, after the second drug entity "etretinate" is recognized, the disease list ***ds*** is iterated again, with one N/Y action being made since there is only one disease entity. The relevant state transitions are shown in Figure 2.

For fair comparison with the baseline, we use Equation 1 to compute the probability of each action, and Equation 2 as the loss function. The training procedure is similar with that in Section 3.3, but the main difference is that the joint model has only one parameter matrix **W**, which is optimized during the joint learning procedure, while the baseline pipeline model has two subtask parameter matrixes trained separately.

4.1 Decoding

Algorithm 1 shows pseudocode of the greedy decoding algorithm, which is a algorithmic description of the state transition system above. The algorithm processes every word of a raw sentence from left to right. It firstly recognizes drug/disease entity mentions based on the current action "*t*",

Algorithm 1 Decoding algorithm of joint models.

```
Input: A sentence s with words, $s_1$, $s_2$, ..., $s_N$.
Output: A set p of drug/disease entities and ADEs in this sentence.
 1: previous action lt, current action t
 2: for i = 1 to N do
 3:     lt ← t
 4:     // recognize drug/disease entity mentions
 5:     t ← GETACTION($s_i$, s, p)
 6:     if t = BC then
 7:         currentEntity ← NEWDRUG($s_i$, s)
 8:         ADDENTITY(p, currentEntity)
 9:     else if t = BD then
10:         currentEntity ← NEWDISEASE($s_i$, s)
11:         ADDENTITY(p, currentEntity)
12:     else if t = I then
13:         currentEntity ← GETLASTENTITY(p)
14:         APPENDWORD(currentEntity, $s_i$, s)
15:     // extract ADE relations
16:     if ISENTITYEND(lt, t) then
17:         for each entity before currentEntity do
18:             lt ← t
19:             t ← GETACTION(entity, currentEntity, s, p)
20:             if t = Y then
21:                 ADDADE(p, entity, currentEntity)
22: return p
```

namely BC, BD, I or O, which is given by "GETACTION" (line 5) considering the current state. Then the algorithm tries to extract ADEs immediately after an entity mention has been fully recognized. "ISENTITYEND" uses "*lt*" and "*t*" to decide the recognition of a full mention. For example, if "*lt*" is BC and "*t*" is O, a drug entity has been recognized. After that, ADEs are extracted by traversing previous entity mentions, pairing them with the newly-discovered entity mention, and deciding whether each drug/disease pair has an ADE relation using Y and N actions, which are given by "GETACTION" (line 19). Note that if "GETACTION" gives an incorrect action (e.g., a BC action when extracting ADEs), it will be ignored. In addition, "GETACTION" (line 5) only extracts features for drug/disease recognition while "GETACTION" (line 19) only extracts features for ADE extraction.

4.2 Global Features

The joint model can exploit interactions between features of the submodels, which is impossible for the pipeline method. We design global features to further exploit this advantage, as shown in Table 2, which can be used by the joint model on top of the **local** features in Table 1 used by the baseline.

The feature templates 1-4 and 8 in Table 2 capture relations between a word or drug/disease pair and extracted ADEs, which can be an important clue for recognizing entities. The first feature template directly leverages word information for a partial or full match to the entities of ADEs. The second and third feature templates conduct a fuzzy match using Brown clusters or WordNet classes. The fourth template can be a guidance when a neighboring entity mention of the current word is contained in an ADE. The eighth template makes use of the previous ADEs for extracting new ADEs.

The feature templates 5-7 capture coordinating relations, which can be important features for inferring ADE relations.

Features for drug/disease recognition
1. whether the current word is contained in an extracted ADE
2. whether the Brown cluster of the current word is identical to that of the word contained in an extracted ADE
3. whether the synonym of the current word is identical to that of the word contained in an extracted ADE
4. whether the entity before the current word is contained in an extracted ADE
5. whether the current word has a coordinating relation with an entity contained in an extracted ADE
Features for ADE extraction
6. whether three entities form the pattern, "A ...... B and C", and the pair A-B is an ADE
7. whether three entities form the pattern, "A and B ...... C", and the pair A-C is an ADE
8. whether a drug/disease pair is contained in an extracted ADE

Table 2: Global features of the joint models.

Taking template 6 for example, in "**Hepatitis**$_{disease}$ caused by **methotrexate**$_{drug}$ and **etretinate**$_{drug}$.", there is a coordinating relation between "methotrexate" and "etretinate". If "Hepatitis" and "methotrexate" have been recognized as an ADE, "etretinate" should also be associated with "Hepatitis". Template 5 and 7 use coordinating relations in various forms.

5 Neural Joint Model

The combination of subtask search candidates can lead to increased feature sparsity in the discrete joint model. A neural model adopts low-dimensional dense features, and can be useful in mitigating this problem. We exploit a neural version of the joint model by replacing the discrete features in our joint model with neural features.

The framework of neural joint model is similar to the transition-based deterministic neural parser of Chen and Manning [2014], as shown in Figure 3. Both models use feed-forward neural network classifiers to perform greedy transition-based decoding. However, there are two main differences: First, we use a convolutional neural network (CNN) to represent variable-length features such as multi-word entity mentions, while their features have fixed lengths. Second, we leverage rectified linear units (RELU) in the hidden layer, while they use a cube activation function.

5.1 Input layer

The input layer consists of a set of embeddings, which replaces the features listed in Table 1 and Table 2. As mentioned in Section 4.1, when recognizing drug/disease entity mentions, only features for drug/disease recognition are extracted while features for ADE extraction are not. Conversely, features for drug/disease recognition are not extracted when extracting ADEs.

Simple features such as words or POS tags can be directly represented as fixed-length embeddings [Chen and Manning, 2014], while variable-length entity features cannot be directly represented like this. We use a CNN to transform them to

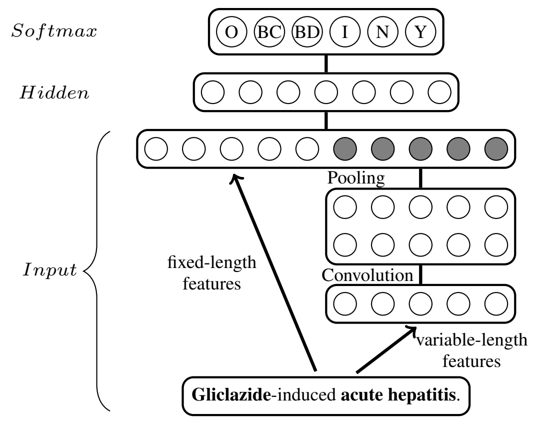

Figure 3: The neural joint model.

fixed-length embeddings. Given an entity **e**=e_1, e_2, ..., e_N, $e_i \in \mathbb{R}^D$ is a D-dimensional word vector corresponding to the i-th word in the entity. In a convolution layer, a kernel with a parameter matrix $\mathbf{W}_1 \in \mathbb{R}^{H_1 \times CD}$ and bias vector $\mathbf{b}_1 \in \mathbb{R}^{H_1}$ convolves the C continuous words $\mathbf{e}_{i:i+C-1}$ from e_1 to e_N. Here H_1 is the output unit number of the kernel. To process all the words in an entity mention, the kernel needs K (K = N-C+1) separate convolutional actions. For each convolution k, the kernel output $\mathbf{m}_k$ is computed as follows:

$$\mathbf{m}_k = tanh(\mathbf{W}_1\mathbf{e}_{i:i+C-1} + \mathbf{b}_1), \tag{3}$$

where D, C and H_1 are hyper-parameters of the model.

The final outputs of CNN are computed through a max-pooling layer by:

$$f_j = \max_{1 \leq k \leq K} m_{kj}, \tag{4}$$

where j indicates the j-th output of the pooling layer, namely the j-th component of the output vector of CNN.

Finally, the input layer $\mathbf{x} \in \mathbb{R}^G$ consists of a concatenation of vectors [**s** : **f**], where **f** indicates output vector of CNN and **s** denotes other fixed-length feature embeddings. G is the total unit number of the input layer.

5.2 Hidden Layer

The hidden layer makes use of a RELU activation function, which is defined as:

$$relu(z) = \max(0, z) \tag{5}$$

Given the input vector **x**, the hidden layer makes a non-linear combination, which can be formulated as:

$$\mathbf{h} = relu(\mathbf{W}_2\mathbf{x} + \mathbf{b}_2), \tag{6}$$

where $\mathbf{W}_2 \in \mathbb{R}^{H_2 \times G}$ is a parameter matrix and $\mathbf{b}_2 \in \mathbb{R}^{H_2}$ is a bias vector. H_2 is a hyper-parameter, denoting number of nodes in the hidden layer.

5.3 Output Layer

The output layer calculates the probabilities of transition actions, so that the one with the maximum probability is selected. The probability of an action a_i is computed by:

$$p(a_i) = softmax(a_i) = \frac{e^{\mathbf{w}_{3_i}\mathbf{h}}}{\sum_{j=1}^{A} e^{\mathbf{w}_{3_j}\mathbf{h}}}, \tag{7}$$

which is similar with that of the baseline max-entropy model. Here A is the number of all possible actions given the state, and $\mathbf{w}_{3_i}$ denotes the i-th row of parameter matrix $\mathbf{W}_3 \in \mathbb{R}^{A \times H_2}$.

For training the neural joint model, Equation 2 is used as the loss function. Mini-batched AdaGrad for neural models [Duchi *et al.*, 2011] with dropout [Hinton *et al.*, 2012] is used to optimize the training objective.

6 Experiments

6.1 Experimental Settings

Data: We use the ADE corpus [Gurulingappa *et al.*, 2012], which consists of 1644 PubMed abstracts for evaluation. Sentences in the corpus are divided into two categories, namely 6821 sentences which contain at least one ADE drug/disease pair (i.e., ADE sentences), and 16695 sentences which contain no ADEs. Only ADE sentences annotated with drug/disease mentions are used in our experiments because we need to evaluate the performance of both entity recognition and relation extraction. We evaluate all the models using 10-fold cross-validation, where 10% of the data are used as the development set, 10% as the test set and the remainder for training.

Metrics: Standard precision (P), recall (R), F_1-measure (F_1) are used for evaluation. An entity is counted as true-positive only if both its boundary and type are correct. An ADE relation is counted as true-positive only if both the boundaries and the types of its entities are correct.

Parameters: For all the models, we set the initial AdaGrad learning rate α and regularization parameter β to 0.01 and 10^{-8}, respectively. For the neural models, embeddings are randomly initialized in the range (-0.01, 0.01), and we set the dimension D to 200 by default. The dropout rate is 0.5. The window size C of the CNN filter is 2 and the size H_1 of the CNN output layer is 200. The hidden layer size H_2 is 200. As it is infeasible to perform full search for all parameters, the values are chosen empirically following prior work on neural networks [Chen and Manning, 2014; Zhang *et al.*, 2015].

Preprocessing: The Stanford CoreNLP toolkit[7] is utilized for preprocessing, such as POS tagging. All the letters are transformed into lowercase forms.

6.2 Development Results

In Table 3, the performance of the discrete joint model with only local features is slightly better than that of the baseline, showing the effectiveness of error propagation reduction. When the global features are added, the performance of the discrete joint model is improved significantly. This demonstrates the important roles of the global features by integrating entity mention and ADE information, which is enabled by the joint model, and not feasible for the baseline model. The neural joint model achieves the best development F_1 scores, improving recalls drastically with a slight decrease in precisions.

[7]http://stanfordnlp.github.io/CoreNLP/

Method	Entity Recognition			ADE extraction		
	P	R	F_1	P	R	F_1
Baseline	78.0	71.8	74.8	60.8	51.6	55.8
Discrete Joint (local)	78.0	72.0	74.9	60.9	51.9	56.0
Discrete Joint (+global)	**80.5**	75.2	77.8	**65.7**	57.0	61.1
Neural Joint	79.6	**79.7**	**79.7**	64.4	**63.3**	**63.9**

Table 3: Performance (%) on the development set.

Method	Entity Recognition			ADE extraction		
	P	R	F_1	P	R	F_1
Li *et al.* [2015]	75.9	71.6	73.6	55.2	47.9	51.1
Baseline	77.8	72.0	74.8	60.7	51.5	55.7
Discrete Joint	**80.0**	75.1	77.5	**65.1**	56.7	60.6
Neural Joint	79.5	**79.6**	**79.5**	64.0	**62.9**	**63.4**

Table 4: Performance (%) on the test set.

6.3 Final Results

There are two main related methods. Kang *et al.* [2014] proposed a knowledge-based method using the Unified Medical Language System. Their results cannot be directly compared with ours since their experimental data were partitioned differently from ours and they did not distinguish the ADE relations from drug-disease-treatment relations. Li *et al.* [2015] proposed a joint model using the perceptron algorithm to extract drug/disease mentions and ADEs simultaneously. Our experimental settings are similar with theirs, and we compare our results on the test set with their best reported results.

As shown in Table 4, the performance of our pipelined baseline is competitive to that of Li *et al.* [2015]. Our discrete joint model gives the best precisions, and the neural joint model gives the best recalls and F_1 scores. Compared with Li *et al.* [2015], our neural joint model improves the F_1 scores in drug/disease recognition and ADE extraction by about 6% and 12%, respectively. The final results demonstrate the advantages of the neural model — on the one hand, two submodels in the joint model can facilitate each other by making use of their interactions and combining features; on the other hand, dense neural features can be effective for overcoming the problem of feature sparsity.

7 Discussion

We compare the baseline with the discrete and neural joint models by analyzing their error distributions. 10% of the test data are randomly selected as the analysis data. We divide the errors into false-positives (FP) and false-negatives (FN), where an entity is counted as false-positive if its boundary or type is incorrectly identified, and an ADE relation is counted as false-positive if its related entity mention boundaries or types are incorrect. An entity or ADE is counted as false-negative if it has not been recognized.

The statistics are shown in Table 5, where the discrete joint model gives the least false-positives and the neural joint model gives the least false-negatives. This is consistent with the overall results in Table 4. Compared with the baseline, one of the advantages of the discrete joint model is that it can utilize extracted ADEs for better recognizing entities. For instance, in "**Gabapentin**$_{drug}$ has been previously reported and usually consists of **anxiety**$_{disease}$, **diaphoresis**$_{disease}$, and **palpitations**$_{disease}$", the last disease "palpitations" is recognized by the discrete joint model but not by the baseline, because the ADE information between "Gabapentin" and "diaphoresis" can be exploited by the joint model. In the discrete joint model, the global features of coordinating relations are also effective. For example, in "an interaction between **clarithromycin**$_{drug}$ and **isradipine**$_{drug}$, potentially increasing the **hepatic toxicity**$_{disease}$", the ADE between "isradipine" and "hepatic toxicity" is successfully extracted by the joint model but not by the baseline.

Method	Entity Recognition		ADE extraction	
	FP	FN	FP	FN
Baseline	229	286	253	324
Discrete Joint	**203**	275	**226**	311
Neural Joint	211	**229**	248	**278**

Table 5: Error analysis using randomly selected data.

The discrete joint model achieves slightly higher precision compared with the neural joint model. One important reason is that its features consist of discrete word patterns, which fire only when exact matches occur. However, this leads to lower recall. One of the reasons for the lower false-negatives of the neural model can be that it can recognize more complex and ambiguous entities. For example, an abbreviation "NMS" (neuroleptic malignant syndrome) can be recognized by the neural model but not by the discrete model. A chemical "6-thioguanine" with mixed digits and letters can also be recognized correctly. More recognized entities give the neural model more opportunities to extract ADE relations.

8 Conclusion

We explored joint models to extract drugs, diseases and ADEs simultaneously. Experimental results on a standard benchmark corpus show that they are more effective compared to traditional pipeline models. In addition, when discrete features are replaced by the neural features, the performance is further improved significantly. We found that a crucial reason behind the effectiveness of the neural network model is its improved recall, which is enabled by dense neural features. Our code is publicly available under GPL at: https://github.com/foxlf823/ade.

Acknowledgments

We thank all reviewers for their detailed comments. This work is supported by the State Key Program of National Natural Science Foundation of China (No.61133012), the National Natural Science Foundation of China (No.61373108, No.61170148), the National Philosophy Social Science Major Bidding Project of China (No.11&ZD189), the Key Program of Natural Science Foundation of Hubei (No.2012FFA088), the Chinese Postdoctoral Science Foundation (No.2013M540593, No.2014T70722).

References

[Bengio *et al.*, 2015] Yoshua Bengio, Ian J. Goodfellow, and Aaron Courville. Deep learning. MIT Press, 2015.

[Bohnet and Nivre, 2012] Bernd Bohnet and Joakim Nivre. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 EMNLP*, pages 1455–1465, 2012.

[Chan and Roth, 2011] Y. Chan and D. Roth. Exploiting syntactico-semantic structures for relation extraction. In *ACL*, Portland, Oregon, 2011.

[Chen and Manning, 2014] Danqi Chen and Christopher D. Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 EMNLP*, pages 740–750, October 2014.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 EMNLP*, 2014.

[Collobert *et al.*, 2011] Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *JMLR*, 2011.

[Duchi *et al.*, 2011] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011.

[Finkel and Manning, 2009] Jenny Rose Finkel and Christopher D. Manning. Joint parsing and named entity recognition. In *Proceedings of the NAACL*, 2009.

[Gurulingappa *et al.*, 2012] H. Gurulingappa, A. Mateen-Rajput, and L. Toldo. Extraction of adverse drug effects from medical case reports. *J. Biomed. Semantics*, 3, 2012.

[Hinton *et al.*, 2012] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. In *CoRR*, 2012.

[Kandula and Treitler, 2010] S. Kandula and Q. Treitler. Exploring relations among semantic groups: a comparison of concept co-occurrence in biomedical sources. *Stud. Health Technol. Inform.*, 2010.

[Kang *et al.*, 2014] N. Kang, B. Singh, C. Bui, Z. Afzal, E. M. Van-Mulligen, and J. A. Kors. Knowledge-based extraction of adverse drug events from biomedical text. *BMC Bioinformatics*, 15, 2014.

[Kummerfeld *et al.*, 2015] Jonathan K. Kummerfeld, Taylor Berg-Kirkpatrick, and Dan Klein. An empirical analysis of optimization for max-margin nlp. In *Proceedings of the 2015 EMNLP*, pages 273–279, September 2015.

[Li and Ji, 2014] Qi Li and Heng Ji. Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd ACL*, pages 402–412, June 2014.

[Li *et al.*, 2015] Fei Li, Donghong Ji, Xiaomei Wei, and Tao Qian. A transition-based model for jointly extracting drugs, diseases and adverse drug events. In *2015 IEEE International Conference on BIBM*, pages 599–602, 2015.

[Liu *et al.*, 2014] Mei Liu, Ruichu Cai, Yong Hu, Michael Matheny, Jingchun Sun, Jun Hu, and Hua Xu. Determining molecular predictors of adverse drug reactions with causality analysis based on structure learning. *Journal of the AMIA*, 21(2):245–251, 2014.

[Nivre, 2008] Joakim Nivre. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34, 2008.

[Roth and Yih, 2007] D. Roth and W. Yih. Global inference for entity and relation identification via a linear programming formulation. *Introduction to Statistical Relational Learning*, 2007.

[Song *et al.*, 2014] Linfeng Song, Yue Zhang, Kai Song, and Qun Liu. Joint morphological generation and syntactic linearization. In *Proceedings of the AAAI Conference*, 2014.

[Wei *et al.*, 2015] Chih-Hsuan Wei, Yifan Peng, Robert Leaman, Allan Davis, Carolyn Mattingly, Jiao Li, Thomas Wiegers, and Zhiyong Lu. Overview of the biocreative v chemical disease relation task. In *Proceedings of the Fifth BioCreative Workshop*, pages 154–166, September 2015.

[Xu and Wang, 2014] R. Xu and Q. Wang. Automatic construction of a large-scale and accurate drug-side-effect association knowledge base from biomedical literature. *J. Biomed. Inform.*, 51:191–199, 2014.

[Yates *et al.*, 2015] Andrew Yates, Nazli Goharian, and Ophir Frieder. Extracting adverse drug reactions from social media. In *Proceedings of the AAAI Conference*, 2015.

[Yildirim *et al.*, 2013] Pinar Yildirim, Ilyas Ekmekci, and Andreas Holzinger. On knowledge discovery in open medical data on the example of the fda drug adverse event reporting system for alendronate (fosamax). In *Springer Lecture Notes in Computer Science*, pages 195–206, 2013.

[Yildirim *et al.*, 2014] Pinar Yildirim, Ljiljana Majnari, Ozgur Ekmekci, and Andreas Holzinger. Knowledge discovery of drug data on the example of adverse reaction prediction. *BMC Bioinformatics*, 15(6), 2014.

[Zhang and Clark, 2008] Yue Zhang and Stephen Clark. Jointword segmentation and pos tagging using a single perceptron. In *Proceedings of ACL*, pages 888–896, 2008.

[Zhang and Clark, 2011] Yue Zhang and Stephen Clark. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37, 2011.

[Zhang *et al.*, 2012] Meishan Zhang, Wanxiang Che, Ting Liu, and Zhenghua Li. Stacking heterogeneous joint models of chinese pos tagging and dependency parsing. In *Proceedings of COLING 2012*, pages 3071–3088, 2012.

[Zhang *et al.*, 2015] Meishan Zhang, Yue Zhang, and Duy-Tin Vo. Neural networks for open domain targeted sentiment. In *Proceedings of the 2015 Conference on EMNLP*, pages 612–621, September 2015.

[Zhou *et al.*, 2005] GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd ACL*, 2005.