Published in final edited form as:
Proc Mach Learn Res. 2018 May ; 90: 57–65.

UArizona at the MADE1.0 NLP Challenge

Dongfang Xu[#], **Vikas Yadav**[#], and **Steven Bethard**
School of Information, University of Arizona, USA

[#] These authors contributed equally to this work.

Abstract

MADE1.0 is a public natural language processing challenge aiming to extract medication and adverse drug events from Electronic Health Records. This work presents NER and RI systems developed by UArizona team for the MADE1.0 competition. We propose a neural NER system for medical named entity recognition using both local and context features for each individual word and a simple but effective SVM-based pairwise relation classification system for identifying relations between medical entities and attributes. Our system achieves 81.56%, 83.18%, and 59.85% F1 score in the three tasks of MADE1.0 challenge, respectively, ranked amongst the top three teams for Task 2 and 3.

Keywords

Adverse Drug Event; Information Extraction; Neural Network

1. Introduction

Adverse drug events (ADEs) are dangerous problems which may lead to unexpected outcome and death in severe cases. According to the report from Agency of Healthcare Research and Quality, ADEs are the main type of nonsurgical adverse event occurring in hospitals in the United States, with an estimated 1.6 million events in 2010(Agency for Health care Research and Quality). Patients hospitalized with an ADE have an increased length of stay, higher costs, and increased risk of in-hospital death compared with those not experiencing an ADE (Poudel et al., 2017). It is commonly accepted that the progress in pharmacovigilance depends on the analysis of ADE-related information from different data sources, especially from electronic health records (EHRs). Employing natural language processing (NLP) techniques on electronic health records (EHRs) provides an effective way of real-time pharmacovigilance and drug safety surveillance.

The shared task MADE1.0 hosted by University of Massachusetts Medical School aims to promote advanced techniques to detect medication and ADEs from EHRs. They annotated 1092 EHR notes with medications, as well as relations to their corresponding attributes, indications and adverse events in Bioc format. MADE1.0 challenge defines three tasks: Task 1 Named entity recognition (NER), Task 2 Relation identification (RI), and Task 3 Integrated task (IT). Similar to the three tasks in MADE1.0 challenge, ADEs extraction is always decomposed into two subtasks, NER and RI. In biomedical named entity recognition tasks, deep learning has yielded numerous state-of-the-art results. Such deep learning systems include Bi-directional Long Short Term Memory and Conditional Random Field (LSTM-CRF) model in Jagannatha and Yu (2016b), and a hybrid system integrating character-based bi-directional LSTM into the word-level LSTM-CRF model (Gridach, 2017). To mitigate the limited data issue, Lee et al. (2017) transfer a neural network (NN) model trained on a large labeled dataset (MIMIC) to another dataset with a limited number of labels which improves the state-of-the-art results on i2b2 2014 and i2b2 2016 datasets. The relation identification task in ADE extraction usually involves identifying complex nary relations, for instance, drugs can have multiple adverse effects simultaneously. In the i2b2 2010 shared task, the No.1 ranked system used an SVM classifier to approach the relation identification task as a pairwise relation classification problem (Roberts et al., 2010). Instead of using pairwise relation classifiers, McDonald et al. (2005) propose to create a graph from pairs of entities that are likely to be related, and then score maximal cliques in that graph as potential complex relation instances. Several recent works adopt the non-pipeline approach, using joint models to solve the two subtasks simultaneously (Riedel and McCallum, 2011; McClosky et al., 2012).

To address the MADE1.0 ADE NLP challenge, we design two independent systems for task1 and task2, respectively. The integrated task is approached by running the two systems sequentially, using the output of the former as input to the latter. The paper is organized as follows: we first present how we preprocess the documents in Section 2, and then explain the NER model for task 1 in Section 3. The RI system is explained in Section 4, and results and analysis are presented in Section 5.

2. Text Preprocessing

Text preprocessing is one of the most important step for information extraction, Akkasi et al. (2016) specifically show the effects of tokenization on the final performance of an NER system on chemical and biomedical text. Effects of encoding techniques on NER performance was highlighed in Cho et al. (2013). We first use the NLTK sentence tokenizer (Bird and Loper, 2004) to segment the paragraphs into sentences and then use the NLTK regexp tokenizer (Bird and Loper, 2004) to tokenize sentences into words.

Our preprocessing code for segmenting sentences into tokens included specific rules for certain cases such as 2mg, 5days, nontender,Noncontributory, etc., where each token is further segmented, for example, 2mg is segmented into 2 and mg, and Noncontributory is splitted into Non and contributory. We do not use any external resources for segmentation but results may vary with changes in segmentation technique as highlighted by Akkasi et al. (2016). Further, to make the most of pre-trained word embedding resources, we lowercase the words for finding its corresponding word embeddings, but for extracting the characters and affix feature, words are taken in their original form without lowercasing so that the word shape information is kept. The words not found in the word embedding vocabulary are assigned the word embedding of the unknown token (assigned as UNK). To further reduce the vocabulary size, the numerical characters and words are replaced by a single token named NUM.

3. NER System

NER is a type of sequence tagging task where each piece of a medical entity is assigned a label that identifies the medical entity that it evokes. We express such labels using the BIO tagging system, where B stands for the beginning of an annotation, I for the inside, and O for outside any annotation. We do not consider the multi-label cases where a single word is assigned to more than 1 tag in this version of the NER system since less than 1.0% of the entities in the entire training dataset have the same offset.

3.1. Neural Architecture

RNNs are the state-of-the-art on sequence tagging tasks (Lample et al., 2016; Graves et al., 2013), thanks to their ability to make predictions conditioned on long distance features, so we also adopt them here. Since many medical entities have special morphological and orthographic information, we want input representations that are sensitive to the spelling of words. As such, our NER system uses the base model of Lample et al. (2016) where we exploit both word context features and word composition with characters using RNNs. In this work, we use LSTM recurrent units in our RNN model, since LSTMs are capable of learning long-term dependencies as well as solving the vanishing gradient problem.

Figure 1 describes the architecture of our model. It first converts the input into features that feed into embedding layers. In the embedding layer, each feature is mapped to a dense vector, and such dense vectors including embeddings for the prefix, suffix, the word itself, and the character-level representation, are then concatenated to form the final representation of the word. The vector representation of each individual word from the embedding layer is then fed into a bi-directional LSTM layer to allow access to both past (left) and future (right) context information. The output of the Bi-LSTM is then given to a CRF layer which outputs one label for each input. The reason for using a CRF layer is that it considers the correlations between labels in neighborhoods and jointly decode the best chain of labels for a given input sequence.

3.2. Input Representation

The input vector representation is generated by concatenating a word embedding, prefix embedding, suffix embedding, and character-level word representation:

- Word Embedding: we use the skip-gram word embeddings trained through a shallow neural network provided by the shared task organizers (Jagannatha and Yu, 2016b,a).
- Prefix and Suffix embedding: we utilize the sub-word affixes from the start and at the end of the word to explicitly provide sub-word information. Yadav et al. (2018) show that what the model learns about affixes is complementary to a recurrent layer over characters, and the usage of affix features in the model improves the performance for the NER task. We select n-gram prefixes and suffixes of words having frequency above a specific threshold to approximate frequent prefixes and suffixes as morphemes of a language.

- Character-level word representation: we use a Bi-LSTM based feature extractor to produce character-level word representations, as shown in Figure 2. Characters of a word are fed into an embedding layer to generate a representation for each character, and the output of the embedding layer is then fed as the input to a Bi-LSTM layer to generate a word-level representation.

Both character and affix embeddings are randomly initialized.

3.3. Network Training

We use the following hyper-parameters: the embedding size of the character, word, prefix, and suffix features are 50, 300, 30 and 30, respectively; the size of the LSTM units in the character-level word representation feature extractor is set to 25; to avoid overfitting, we use dropout with probability 0.60 for the NER embedding layer (applied after concatenating word embedding, character-level word representation and affix embeddings); we trained the model with Stochastic Gradient Descent (SGD) on mini-batches of size 50, and set the learning rate, and learning decay rate as 0.10, and 0.99, respectively. We implement our model in tensorflow and run the model on the El-Gato supercomputer at the University of Arizona, and the model is trained for 150 epochs on the entire training dataset.

4. RI System

Given the NER annotations, the RI system aims to extract 7 well-defined relations between Medical Attributes and their relevant Medical Entities. Note that the medical entity and its associated attribute may not appear in the same sentence or even paragraph, and that each medical attribute may link to zero or more medical entities. Considering the facts in the dataset, we build a simple but effective system to approach the task as 7 independent pairwise relation classification problems, one for each relation type.

4.1. Generate Entity-Attribute Pairs

For each medical attribute, we obtain a set of medical entity candidates that may participate in a relation using the rules that 1) medical entities appear within a 3-entity window of medical attributes, for example, all *Drug* entities appearing within a 3-*Drug* window of the attribute *Frequency* would be considered as candidates; 2) the distance in number of characters between the attribute and entity candidate is smaller than 1000. The generation of the entity-attribute pairs is liberal, covering more than 97% of the positive pairs, while still filtering out infrequent negative ones, thus mitigating the imbalanced class issues of the entity-attribute pairs.

4.2. Features

The relation classifiers use 4 types of features to predict binary output for each entity-attribute pair:

- Position: the position of the entity candidate with respect to the attribute among the entire entity candidates of the attribute, where the position of medical attribute is set to 0. The position of the entity candidate ranges from −3 to +3.
- Distance: the distance in number of characters and words between the entity pair.

- Bag of Words: all words within a 10-word window before and after the entity and attribute, plus the entity and attribute texts. We retained as features only the 903 words that appeared in such context windows with frequencies ≥500 across the entire dataset. Thus, for each entity pair we generated 903 bag-of-word features: the counts of how many times each unique word appears in the context.
- Bag of Entities: the counts of all annotation types between the entity and attribute.

4.3. Learning Model

For each entity-attribute relation classifier, we trained a support vector machine using C-Support Vector Classifier (Chang and Lin, 2011) in scikitlearn python package. We experimented with multiple kernels and selected the radial basis function with the kernel coefficient, γ, and the penalty parameter, C, set to their defaults. We tuned the class weight for each relation classifier for the best performance in 5-fold cross validation. Other parameters were set to their defaults.

5. Results & Discussion

Training and evaluation of UArizona system utilizes the 1092 de-identified EHR notes from 21 cancer patients provided by the task organizers. The results of Task-1 NER on the test dataset are reported in Table 1. We find that for drug entity attributes *Drug* (drug name), *Frequency*, *Route*, *Dosage*, and *Duration*, our model works much better than the remaining medical entities. For example, the model obtains 87.55% F1 score for *Drug* identification in strict evaluation, which is the second highest score among all other entities, while for the *ADE* and *Indication* (called the medical symptom entities), the model only gets 55.07% and 60.23% F1 in strict evaluation. The performance differences between these two different entity types could be attributed to the annotation distributions in the dataset, i.e., there are much more drug entity attributes than medical symptom entities, and the tokens annotated as medical symptom entities are much more diversified than tokens annotated as drug entity attributes. It is also notable that the identification score 81.93% for entity *SSLIF* is much higher than other medical symptom entities *ADE* and *Indication*, since *ADE* only refers to the medical signs or symptoms resulting from the normal use of a drug and *Indication* only refers to the symptoms being actively treated, without using external knowledge like medical ontology, it is difficult for the model to make the inferences by using the word context and local features alone.

Table 2 shows the results of Task-2 RI and Task-3 IT on the test dataset. Since the organizers did not release the complete test dataset, we can only report the F1 score for task 2 and 3 here. By using the RI system alone on the gold identified entities, the system achieves the overall F1 of 83.18%. And when integrating both NER and RI systems for task-3, our system obtains 59.85% F1 score. At the step of generating entity-attribute pairs in RI system, we narrow down the scope by adding constraints such as distance rule, which could increase the precision, but also ignore the long-term dependency, and thus resulting in low performance for extracting *adverse* and *reason* relations.

6. Conclusion and Future Work

Our system is currently amongst the top three teams for Task 2 and 3 in the MADE 1.0 challenge, but there are still many improvements that can be made. Notably, we do not use any external resources except the pre-trained word embedding in our system, we believe that by using existing knowledge resources, such as SNOMED-CT, our system could be more robust and accurate on this ADEs task. We also plan to expand our use of neural models to the RI task, and implement a joint model to extract both entities and relations simultaneously.

Acknowledgments

This work was supported by National Institutes of Health grant R01GM114355 from the National Institute of General Medical Sciences (NIGMS). The computations were done in systems supported by the National Science Foundation under Grant No. 1228509. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health or National Science Foundation.

References

Agency for Health care Research and Quality. Hospital-acquired conditions update: Interim data from national efforts to make care safer, 2010–2014. https://www.ahrq.gov/sites/default/files/publications/files/interimhacrate2014_2.pdf. Accessed: 2018-04-21.

Akkasi Abbas, Varoğlu Ekrem, and Dimililer Nazife. Chemtok: a new rule based tokenizer for chemical named entity recognition. BioMed research international, 2016, 2016.

Bird Steven and Loper Edward. Nltk: the natural language toolkit In Proceedings of the ACL 2004 on Interactive poster and demonstration sessions, page 31 Association for Computational Linguistics, 2004.

Chang Chih-Chung and Lin Chih-Jen. Libsvm: a library for support vector machines. ACM transactions on intelligent systems and technology (TIST), 2(3):27, 2011.

Cho Han-Cheol, Okazaki Naoaki, Miwa Makoto, and Tsujii Junichi. Named entity recognition with multiple segment representations. Information Processing & Management, 49(4):954–965, 2013.

Graves Alex, Mohamed Abdel-rahman, and Hinton Geoffrey. Speech recognition with deep recurrent neural networks In Acoustics, speech and signal processing (icassp), 2013 ieee international conference on, pages 6645–6649. IEEE, 2013.

Gridach Mourad. Character-level neural network for biomedical named entity recognition.Journal of biomedical informatics, 70:85–91, 2017. [PubMed: 28502909]

Jagannatha Abhyuday N and Yu Hong. Bidirectional rnn for medical event detection in electronic health records. In Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting, volume 2016, page 473 NIH Public Access, 2016a.

Jagannatha Abhyuday N and Yu Hong. Structured prediction models for rnn based sequence labeling in clinical text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing, volume 2016, page 856 NIH Public Access, 2016b.

Lample Guillaume, Ballesteros Miguel, Subramanian Sandeep, Kawakami Kazuya, and Dyer Chris. Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360, 2016.

Lee Ji Young, Dernoncourt Franck, and Szolovits Peter. Transfer learning for named-entity recognition with neural networks. arXiv preprint arXiv:1705.06273, 2017.

McClosky David, Riedel Sebastian, Surdeanu Mihai, McCallum Andrew and Manning Christopher D. Combining joint models for biomedical event extraction In BMC bioinformatics, volume 13, page S9 BioMed Central, 2012.

McDonald Ryan, Pereira Fernando, Kulick Seth, Winters Scott, Jin Yang, and White Pete. Simple algorithms for complex relation extraction with applications to biomedical ie In Proceedings of the

43rd Annual Meeting on Association for Computational Linguistics, pages 491–498. Association for Computational Linguistics, 2005.

Poudel Dilli Ram, Acharya Prakash, Ghimire Sushil, Dhital Rashmi, and Bharati Rajani. Burden of hospitalizations related to adverse drug events in the usa: a retrospective analysis from large inpatient database. Pharmacoepidemiology and drug safety, 26(6): 635–641, 2017. [PubMed: 28233421]

Riedel Sebastian and McCallum Andrew. Fast and robust joint models for biomedical event extraction In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1–12. Association for Computational Linguistics, 2011.

Roberts Kirk, Rink Bryan, and Harabagiu Sanda. Extraction of medical concepts, assertions, and relations from discharge summaries for the fourth i2b2/va shared task In Proceedings of the 2010 i2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data. Boston, MA, USA: i2b2, 2010.

Yadav Vikas, Sharp Rebecca, and Bethard Steven. Deep a x features improve neural named entity recognizers In Proceedings of The Seventh Joint Conference on Lexical and Computational Semantics (*SEM 2018, to be published). SEM, 2018.

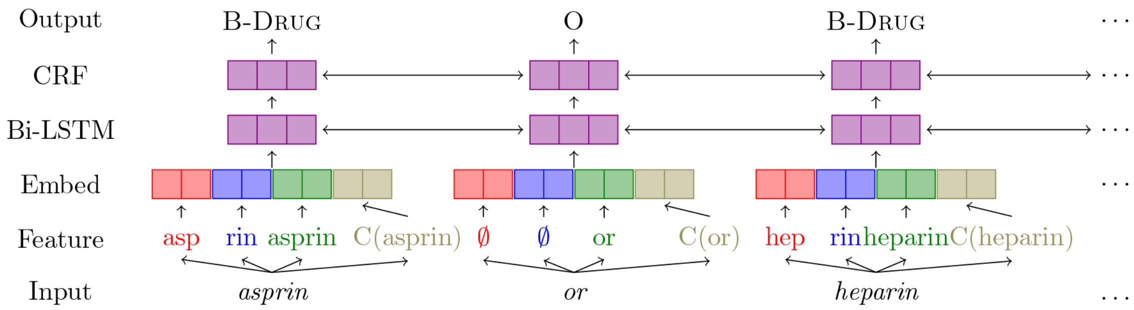

Figure 1:

NER model architecture diagram taken from Yadav et al. (2018). The input is *asprin or heparin*. At the feature layer, *asp* is the prefix, *rin* is the suffix, *C*(*asprin*) is a vector representation generated from characters of *asprin*. If the word doesn't have any subword information, both prefix and suffix are set as∅.

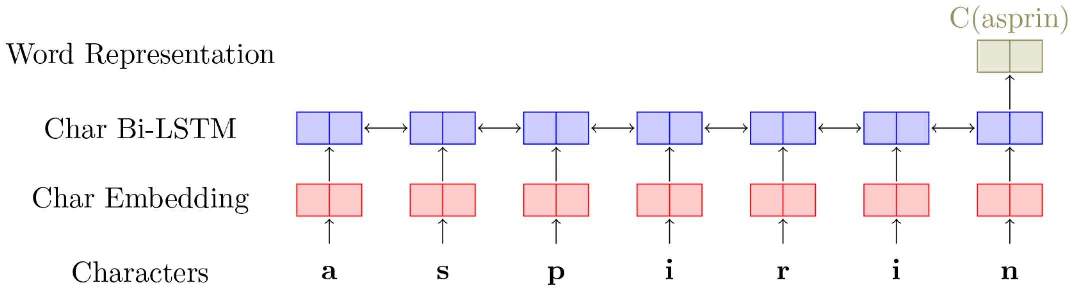

Figure 2:

The Bidirectional-LSTM neural network for extracting character-level representations of words. The input of the neural network is characters of the *asprin*, and the output at the last step of the Bi-LSTM layer is used as the character-level representations of word.

Table 1:

Task −1 NER results: Precision (*P*), recall (*R*), and F_1 of our models on MADE1.0 test dataset using official evaluation script provided by the organizers. Strict evaluation includes exact match of entity boundaries and character offset along with exact match of entity type, while relaxed evaluation is conducted at word level.

Entity type	**Strict Scores**			**Approximate/Relaxed scores**		
	R	**P**	**F1**	**R**	**P**	**F1**
Drug	87.06	88.05	87.55	89.53	92.90	91.18
Indication	58.33	62.25	60.23	58.32	62.96	60.55
Frequency	82.85	87.08	84.91	83.09	90.95	86.85
Severity	74.91	77.52	76.19	80.15	87.96	83.87
Dose	84.02	85.73	84.87	94.80	93.19	93.99
Duration	76.69	78.46	77.57	75.81	81.08	78.36
Route	92.29	92.76	92.53	78.54	81.31	79.90
ADE	42.23	79.13	55.07	41.23	80.11	54.44
SSLIF	82.11	81.74	81.93	82.77	82.80	82.79
Overall	80.42	82.73	81.56	81.34	84.64	82.95

Table 2:

Task 2 RI and Task 3 IT results: F_1 of our models on MADE1.0 test dataset.

Task	**F1**
Task 2 RI	83.18
Task 3 IT	59.85