Drug Safety (2019) 42:113–122
https://doi.org/10.1007/s40264-018-0765-9

ORIGINAL RESEARCH ARTICLE

Adverse Drug Event Detection from Electronic Health Records Using Hierarchical Recurrent Neural Networks with Dual-Level Embedding

Susmitha Wunnava[1] · Xiao Qin[1] · Tabassum Kakar[1] · Cansu Sen[1] · Elke A. Rundensteiner[1] · Xiangnan Kong[1]

Published online: 16 January 2019

Abstract

Introduction Adverse drug event (ADE) detection is a vital step towards effective pharmacovigilance and prevention of future incidents caused by potentially harmful ADEs. The electronic health records (EHRs) of patients in hospitals contain valuable information regarding ADEs and hence are an important source for detecting ADE signals. However, EHR texts tend to be noisy. Yet applying off-the-shelf tools for EHR text preprocessing jeopardizes the subsequent ADE detection performance, which depends on a well tokenized text input.

Objective In this paper, we report our experience with the NLP Challenges for Detecting Medication and Adverse Drug Events from Electronic Health Records (MADE1.0), which aims to promote deep innovations on this subject. In particular, we have developed rule-based sentence and word tokenization techniques to deal with the noise in the EHR text.

Methods We propose a detection methodology by adapting a three-layered, deep learning architecture of (1) recurrent neural network [bi-directional long short-term memory (Bi-LSTM)] for character-level word representation to encode the morphological features of the medical terminology, (2) Bi-LSTM for capturing the contextual information of each word within a sentence, and (3) conditional random fields for the final label prediction by also considering the surrounding words. We experiment with different word embedding methods commonly used in word-level classification tasks and demonstrate the impact of an integrated usage of both domain-specific and general-purpose pre-trained word embedding for detecting ADEs from EHRs.

Results Our system was ranked first for the named entity recognition task in the MADE1.0 challenge, with a micro-averaged F1-score of 0.8290 (official score).

Conclusion Our results indicate that the integration of two widely used sequence labeling techniques that complement each other along with dual-level embedding (character level and word level) to represent words in the input layer results in a deep learning architecture that achieves excellent information extraction accuracy for EHR notes.

Part of a theme issue on "NLP Challenge for Detecting Medication and Adverse Drug Events from Electronic Health Records (MADE 1.0)" guest edited by Feifan Liu, Abhyuday Jagannatha and Hong Yu.

✉ Susmitha Wunnava
swunnava@wpi.edu

Xiao Qin
xqin@wpi.edu

Tabassum Kakar
tkakar@wpi.edu

Cansu Sen
csen@wpi.edu

Elke A. Rundensteiner
rundenst@wpi.edu

Xiangnan Kong
xkong@wpi.edu

[1] Worcester Polytechnic Institute, 100 Institute Rd, Worcester, MA 01609, USA

Key Points

Simple but effective preprocessing strategies aid in improving the overall performance of the system.

Morphological features obtained through character-level representations of the word enhance the overall representation of a word.

Structured prediction models based on conditional random fields help improve the prediction accuracy of named entities that span across multiple words and phrases.

1 Introduction

Adverse drug events (ADEs) are known to be a leading cause of death in the United States [1]. Early detection of ADE incidents aids in the timely assessment, mitigation and prevention of future occurrences of severe, potentially fatal ADEs. Natural language processing (NLP) techniques towards recognizing ADEs and related information from spontaneous reports, clinical reports, and electronic health records (EHRs) provide an effective method for drug safety monitoring and pharmacovigilance.

A major challenge with processing EHR records is that EHR notes, while containing valuable knowledge, correspond to unstructured text. Numerous challenges arise when extracting entities from such narratives. Often the notes contain medical and non-medical abbreviations, acronyms, numbers and misspelled words, which make it difficult to recognize the critical information in the notes. In other words, certain types of information such as ADEs, indications, and signs and symptoms are harder to detect than other information such as drug names. This can be explained by the following. First, these entities can span across multiple words, about one to seven words per entity. Also, some entities could be expressed as a combination of entity-specific medical terms as well as non-medical descriptive text [2]. For instance, in the phrase "coronary artery disease related event prophylaxis," the words "related" and "event" are descriptive text, while the rest are medical terms. Moreover, there is a lot of ambiguity among relevant named entities. Depending upon the context, the same exact phrase can be an ADE, indication, or a sign and symptom (SSLIF). Table 1 states key challenges of textual notes. The example text is taken from a de-identified data set of EHR notes of 21 cancer patients from the University of Massachusetts Medical School.

To tackle these challenges, an ADE detection system should [1] capture both syntactic and semantic features of the words to best distinguish between ADE-related terms and non-medical words, [2] model the dependencies among words within a sentence so that ADR-related phrases consisting of multiple strongly associated words (including non-medical words) can be identified, and [3] master plan the ADE detection by considering the possible labeling outcomes for each word so that the detected ADE words or phrases as a whole in a sentence make sense. Following this principle, we propose our Dual-Level Embedding for Adverse Drug Event Detection (DLADE) framework—a three-layered, deep learning architecture that solves the listed three challenges jointly within one model. In addition, due to the noisy nature of the EHR text data, we design a rule-based EHR text preprocessor for providing clean tokenized text input essential for the success of the subsequently applied computational detection method.

2 Related Work

2.1 Rule-based method

Rule-based extraction techniques are user-created pattern matching rules that require human expertise. In [3, 4] a rule-based approach that combines rules with semantic lexicons is used to extract drugs and related information such as dosage, duration and signs or symptoms from clinical records.

2.2 Machine learning method

In [3, 5] statistical and machine learning techniques such as Hidden Markov Models (HMMs) and conditional random fields (CRFs) are used to extract information from

Table 1 Examples showing key challenges of biomedical text

Challenges	Example
Multiple words	Lymphoplasmacytoid lymphoma involving bone marrow and spleen
Medical and non-medical words	Cervix again is significantly stenotic
Abbreviations	IgG (stands for immunoglobulin G)
Ambiguous named entities	Headaches (indication or ADE or sign or symptom)

ADE adverse drug event

biomedical text. Ramesh et al. [6] developed a machine learning–based biomedical named entity tagger using support vector machines (SVMs), to extract medication and ADE information from medical narratives.

In recent years, deep learning models, especially recurrent neural network (RNN) models, have been shown to be promising techniques for sequence tagging and named entity recognition (NER) tasks due to their ability to learn from the context surrounding the words in a sequence [7]. Long short-term memory (LSTM) [8] is a type of RNN that is effective at learning the long-term dependencies between words in a sequence. CRFs [9] are probabilistic graphical models that have been used for sequence labeling tasks because of their ability to model the dependencies in the outputs of a sequence. A combination of RNN and CRF models have also been explored and found to be effective for sequence tagging [10–12]. Most of the deep learning models developed for NER tasks use word embeddings as an input to the models. Word embeddings are vector representations of words in the text. These word embeddings can either be trained on domain-specific text, such as biomedical texts, EHR notes, and PubMed articles [13, 14], or they can be trained on a wide variety of general text, such as Wikipedia articles [15].

3 The MADE1.0 NLP Challenge

This section provides a brief introduction to the MADE1.0 NLP Challenges for Detecting Medication and Adverse Drug Events from Electronic Health Records hosted by University of Massachusetts at Lowell, Worcester, and Amherst.[1] The main objective of the challenge is to advance ADE detection techniques to improve patient safety and healthcare quality. The challenge consists of the following three tasks: [1] NER, [2] relation identification (RI), and [3] integrated task (IT).

3.1 The Task

We have developed our system DLADE [16] specifically for "Task 1, the Named Entity Recognition (NER) problem" of the challenge. The task is to develop a system capable of automatically detecting any mentions of medication names and their attributes (dosage, frequency, route, and duration) as well as mentions of ADEs, indications, and other signs and symptoms. Tasks 2 and 3 (RI and IT) are beyond the scope of this paper.

3.2 Data Set

The MADE1.0 challenge used a total of 1089 de-identified EHR notes from 21 cancer patients. The notes are annotated with medication information (such as medication name, dosage, route, frequency, and duration), ADEs, indications, and other signs and symptoms. The annotated notes were released in the BioC format [17]. Of these reports, 876 were released to participants of the competition for the development of their learning system along with the gold standard annotation.

3.3 Resources

This challenge restricted the usage of existing NLP tools such as NLTK [18], Stanford NLP [19], and Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES) [20], which should only be used for text preprocessing, in order to assure fairness among competition participants, who included both university as well as company contributors with diverse resource access. The term standard resources refers to the training data released to the participating teams, the pre-trained word embedding trained using wiki, and de-identified Pittsburgh EHR and PubMed articles [10, 21] and the Unified Medical Language System (UMLS) [22]. The term extended resources refers to publicly available tools designed to work with medical concepts and medical relations as well as any ancillary corpus in addition to the standard resources. Our system, DLADE, is developed using only the standard resources released as part of the challenge—the training data and the pre-trained word embedding.

3.4 Evaluation Process for MADE1.0 Challenge

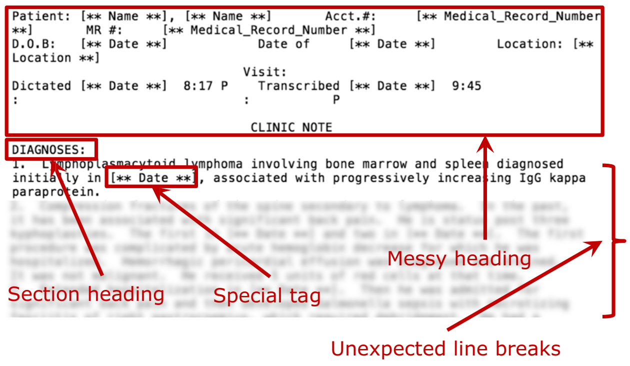
The developed system was then evaluated by the MADE1.0 organizers on two different tracks: [1] the Standard track, using only the standard MADE1.0 resources, and [2] the Extended track, using customized resources available publicly. The top teams for the AMIA 2018 Informatics Summit panel presentation were selected based only on the performance of each team for the Standard track. The evaluation is based on the strict matching in F1-score using exact phrase-level evaluation. Relaxed matching using word-level evaluation is not considered. The metrics used for evaluating the systems are precision, recall, F1-score and the micro-averaged score, which sums up the individual true positives, false positives, and false negatives of the system for different sets and then applies them to get the statistics. The best score is determined by the micro-averaged F1-score for the Standard track using an exact phrase-level evaluation. This simplified method selected a winner for this task of the competition.

4 Our Approach: the DLADE System

4.1 Preprocessing

As we will explain in Sect. 5.2, our model considers the EHR notes as a set of sentences, where each individual sentence

[1] See https://bio-nlp.org/index.php/projects/39-nlp-challenges.



```
Patient: [** Name **], [** Name **]       Acct.#:     [** Medical_Record_Number
**]       MR #:     [** Medical_Record_Number **]
D.O.B:   [** Date **]             Date of     [** Date **]        Location: [**
Location **]
                              Visit:
Dictated [** Date **]  8:17 P     Transcribed [** Date **]  9:45
:                              :           P

                                CLINIC NOTE
DIAGNOSES:
1.  Lymphoplasmacytoid lymphoma involving bone marrow and spleen diagnosed
initially in [** Date **], associated with progressively increasing IgG kappa
paraprotein.
[illegible]
```

Fig. 1 Noise in the electronic health record text

in turn consists of a sequence of words. Therefore, we first tokenize the EHR notes into sentences and then tokenize the words within each sentence. MADE1.0 EHR notes contain noise (Fig. 1), e.g., section headings with repeating punctuation and abnormal text formatting, and unexpected line breaks, where existing off-the-shelf tokenizers such as NLTK [23] fail to produce promising results, which we show in the results section. For this reason, we instead built a rule-based tokenizer that processes the EHR note character by character for sentence (see algorithm 1) and word chunking while concurrently recording the character offsets with respect to the original text file. The rule-based processes the text narrative character by character. It decides the sentence boundary by considering the period sign with additional conditions to avoid false alarms, such as "Dr.", "Mr." and "1.23" etc. The sentences (except the first one, which usually consists of encrypted headings) are then passed to a rule-based word tokenizer which decides the word boundary by considering the spacing among the words. The tokenizers also record the boundary offsets for evaluation purposes. The source code of the preprocessor is released.[2]

Some named entities correspond to multiple words. Hence, we use the inside, outside, beginning (IOB) [24] tagging scheme to distinguish between the beginning of an entity (tag B: named entity) and the inside of an entity (tag I: *named entity*). The no-entity tag is O.

4.2 Word Embedding

Word embeddings are dense representations of words that encode both syntactic and semantic features of the words into a vector. Each word is mapped to a real-valued vector of a low dimensional space, the dimensionality typically ranging between 50 and a few hundred (such as 200 or 300). This is much smaller than that for the one-hot vector representations of the words, the dimensions of which are usually in the thousands with sparse vectors.

Word embeddings have been shown to improve the performance of sequence tagging tasks [25] and are an integral part of deep learning models. Word embeddings can be learned from the training data if the training corpus is large enough and has a good vocabulary size. In this case, the vectors are randomly initialized and passed to the neural network in order to learn and further tune the random vectors to provide a good meaningful representation of the words. Alternatively, there are publicly available pre-trained word embeddings which can be readily passed as input to the deep learning models. Some of these pre-trained word embeddings are trained specifically for a domain or task, such as biomedical text [13, 14], while others are more general purpose and are trained on Wikipedia articles [15]. The pre-trained embeddings can either be fixed while training the network or can be further tuned to make them better representations specific to the task.

Word embeddings can also be learned from the characters in the word. Character-level representations of the word capture the morphological features such as the prefix or the

Algorithm 1 Rule-based Sentence Tokenizer.

```
Sentence Tokenizer (Document):
  For Character in Document:
    If Character is '.' Then:
      If Previous Word's length > k and Previous and Next Character is not a Number:
        Break a Sentence
    Else:
      Continue
```

[2] See https://github.com/qinxiao/DLADE.

suffix of a word, words starting with upper-case letters, and abbreviations, which can be encoded into a dense representation [26]. The character-level embeddings can be used to supplement the learned or pre-trained word-level embeddings to train a deep learning model.

In the context of an ADE detection task, one of the challenges with EHR text (as shown in Table 1) is that it includes various medical terms and abbreviations. The pre-trained word embedding for such words and phrases might not be available, especially if they occur not too frequently in the corpus on which the word embeddings are learned from. In such cases where a pre-trained word embedding is unavailable, the learned character-level representation of the word will enable us to extract the meaning of words. Neural network models such as convolutional neural networks (CNNs) or RNNs can be used to run over the sequence of characters in a word to learn the character-level representation of the word.

4.3 Methods

In this section, we describe the methods used in our system: bi-directional long short-term memory (Bi-LSTM), CRF, and Bi-LSTM-CRF.

4.3.1 Bi-LSTM

RNN models are designed to capture the long-term dependencies in a sequence. They have an input layer, hidden layer and output layer. The input layer takes the word features in the form of word embeddings. The hidden layer maintains information on previous outputs, enabling it to predict the current output based on the past information and previous word in the sequence. The output layer produces the probability distribution for each label. However, RNNs are less effective with longer sequences because of the problem with vanishing or exploding gradient [27, 28], and thus result in a network that cannot learn well from the longer training sequences.

LSTM neural network [5] is a type of RNN that is designed to overcome the gradient vanishing/exploding problem and thus efficiently learn the long-term dependencies in a sequence [29]. They have a built-in memory cell within the hidden layer which is responsible for controlling the flow of previous outputs to the current output without exploding the gradient. However, an LSTM network only captures information about the previous context and does not take into account the future context of the current output.

Bi-LSTM networks [30] have proven to be very useful to capture the entire context by processing the sequence in both forward and backward directions with two hidden layers, one for each direction. The output from both directions is concatenated to form the final output. In the context of an ADE detection task, the sentences in the EHRs are often long sequences including named entities that often span across multiple words within the sequence. The named entities are also heavily dependent upon the context they occur in, and more often the same word or phrase can be tagged as two different named entities depending upon the context.

4.3.2 CRF

CRF models [9] are widely used for sequence labeling tasks. Given a sequence, the model uses contextual information from preceding and succeeding information in the sequence to predict the current label. The models predict the label sequence jointly instead of predicting each label individually. These models can predict sequences where multiple words depend on each other. In the context of ADE detection, one of the challenges with EHR text is that the named entities can occur as a combination of medical and non-medical words. For instance, in the named entity phrase "cervix again is significantly stenotic," the label for each of the words in the phrase is greatly dependent on the label of the previous word.

4.3.3 Bi-LSTM and CRF

LSTM and CRF have their own advantages and disadvantages. LSTM is better for modeling long sequences of words, but the label for each word is predicted independently and not as a part of the sequence. CRF is better for modeling the entire sequence jointly, but needs handcrafted features to obtain significantly good results. A combination of Bi-LSTM and CRF models [7] has been used for sequence tagging where each one of the models contributes to the combined model while complementing each other. In the sequence tagging task, Bi-LSTM is used to capture the contextual representation of the words from the input features. The outputs from the Bi-LSTM are fed to the CRF layer to jointly predict the best label sequence.

4.3.4 DLADE Model

Given the success of deep learning models for NLP tasks [10, 31], we have developed a deep learning–based system that utilizes the combined effectiveness of RNNs, more precisely Bi-LSTM [30] models and CRF by integrating them into one deep network architecture. The Bi-LSTM networks have been widely used for NLP tasks to learn the context representation of a word in a sequence by traversing through the sequence in both forward and backward (i.e., reverse order) directions.

In a nutshell, our model is composed of a Bi-LSTM neural network for an input layer responsible for character embedding and a second Bi-LSTM for word embedding followed by a linear-chain CRF output layer. We have used the pre-trained medical word embedding provided by the MADE1.0 challenge [21, 10]. More precisely, first at the

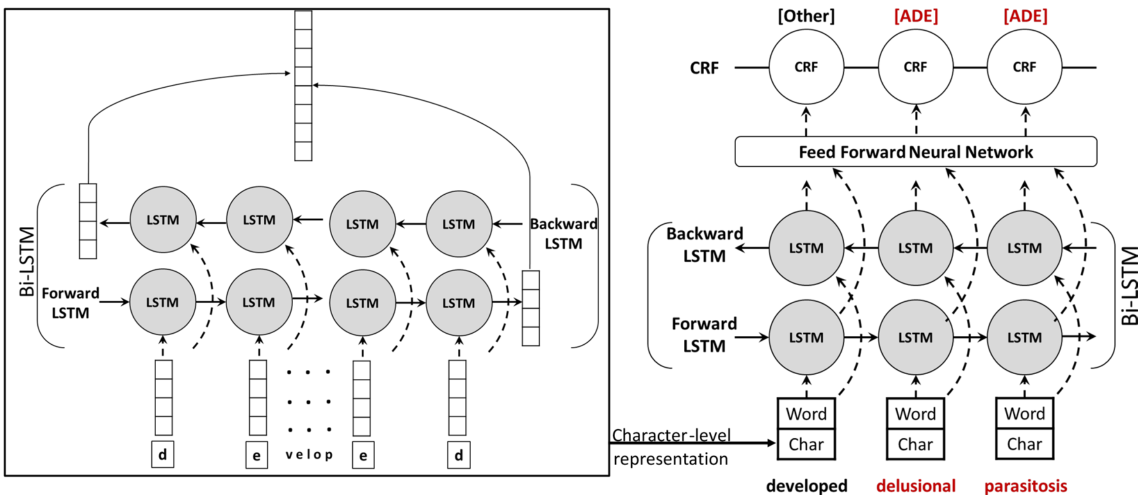


Fig. 2 DLADE system architecture. *ADE* adverse drug event, *Bi-LSTM* bi-directional long short-term memory, *CRF* conditional random field, *DLADE* Dual-Level Embedding for Adverse Drug Event Detection, *LSTM* long short-term memory

Table 2 Evaluation results on the final MADE1.0 holdout test set

	ADE	Dose	Drug	Duration	Frequency	Indication	Route	Severity	SSLIF	**Micro-avg**
Precision	0.7261	0.8721	0.9066	0.7143	0.8438	0.6587	0.9100	0.7798	0.8309	**0.8373**
Recall	0.5644	0.8874	0.9019	0.8271	0.8412	0.6216	0.9381	0.8362	0.8570	**0.8454**
F1-score	0.6351	0.8797	0.9042	0.7666	0.8425	0.6396	0.9239	0.8070	0.8438	**0.8413**

ADE adverse drug event, *avg* averaged, *SSLIF* other sign, symptom or disease

bottom, character-level representations which capture the morphology of a word are computed by running a Bi-LSTM over the sequence of characters in the input words. Consolidated dense embedding, comprising pre-trained medical word embedding concatenated with a learned character-level representation, is used to represent a word. Figure 2 shows our system architecture. Although for the MADE1.0 challenge we have used the MADE1.0 pre-trained word embedding, our system is designed to plug-and-play with any pre-trained word embedding.

We feed this dense embedding of each word into a second Bi-LSTM. This second Bi-LSTM then extracts the contextual representation of each word in the sentence that captures information from the meaning of the word, its characters and its context. The output from the Bi-LSTM is used as the input to a feed-forward neural network to compute a vector of scores, where each entry corresponds to a score for each tag. Tags are the individual named entities. To make the final prediction, the output of the feed-forward network is passed to a linear-chain CRF. The overall model is trained by minimizing the negative log-likelihood.

5 Experimental Results

5.1 Hyperparameter Settings

The named entities are Drug, Indication, Frequency, Severity, Dose, Duration, Route, ADE, and SSLIF (other sign, symptom or disease). The model operates on the tokenized sentences. We used a batch size of 20 sentences. We did not make any restrictions on the sentence length. Rather, we used the maximum length of the sentences in a batch. All shorter sentences in that batch are padded with masks. As input, the pre-trained word embeddings are 200 dimensional vectors and the learned character-level embeddings are 100 dimensional vectors. The hidden state is set to 100 dimensions for running Bi-LSTM for learning character embedding. The hidden state is set to 300 dimensions for running Bi-LSTM with dense word embedding. To avoid overfitting, we applied a dropout strategy [31, 32] of 0.5 in our model. All the models were trained with a learning rate of 0.001 using Adam [33]. Our models are trained on Intel® Xeon® 2.10 GHz, with a total memory of 251 GB. They are implemented using the TensorFlow framework [34].

Table 3 Evaluation results on the final MADE1.0 holdout test set with NTLK tokenizer

	ADE	Dose	Drug	Duration	Frequency	Indication	Route	Severity	SSLIF	Micro-avg
Precision	0.7081	0.8451	0.9066	0.8889	0.8398	0.7024	0.9073	0.7649	0.8206	0.8337
Recall	0.5236	0.8811	0.8988	0.782	0.8563	0.6006	0.9330	0.8723	0.8654	0.8474
F1-score	0.6020	0.8627	0.9027	0.8320	0.8480	0.6475	0.9200	0.8151	0.8424	0.8405

ADE adverse drug event, *avg* averaged, *SSLIF* other sign, symptom or disease

Table 4 Improvement for MADE1.0 in F1-score when using dual-level embedding

	Word embedding	Dual-level (character + word) embedding	Improvement (%)
ADE	0.5848	0.6351	8.6
Dose	0.8172	0.8797	7.6
Drug	0.8780	0.9042	3.0
Duration	0.6879	0.7666	11.4
Frequency	0.7964	0.8425	5.8
Indication	0.6151	0.6396	4.0
Route	0.8705	0.9239	6.1
Severity	0.7648	0.8070	5.5
SSLIF	0.8290	0.8438	1.8
Micro-averaged	0.8147	0.8413	3.3

ADE adverse drug event, *SSLIF* other sign, symptom or disease

5.2 Methodology

Our system DLADE is trained on the 876 EHR notes from MADE1.0. From the training set of sentences, 10% of the sentences are held out as a validation set. This allows us to evaluate the model in the training phase by determining the best F1-score for early stopping. If there is no improvement in the F1-score within the last three consecutive epochs, the system performs an early stopping.

5.3 Results on MADE1.0 Test Data Set

On the evaluation test set consisting of 213 EHR notes, our deep network achieves a micro-averaged precision, recall and F1-score of 0.8373, 0.8454, and 0.8413, respectively, for the exact phrase-level evaluation. Table 2 shows our evaluation results on the MADE1.0 evaluation test set for each of the entities. Our system has been selected as one of the top three performers and ranked first in the MADE1.0 challenge for the Standard NER task.

To demonstrate the effectiveness of our rule-based tokenizer, we compared the prediction results from DLADE, which uses our proposed rule-based tokenizer, with a baseline system that uses the NLTK tokenizer. Table 3 shows that the baseline system achieves higher micro-averaged recall; however, it gets a lower micro-averaged F1-score.

To demonstrate the effectiveness of utilizing dual-level embedding, we compared the prediction results from DLADE, which uses both the learned character-level representations of a word and the pre-trained word-level embedding, with a baseline system that utilizes only the pre-trained word-level embedding.

Table 4 compares the F1-scores of individual entities as well as the overall micro-averaged F1-score of all entities combined. It shows the percentage improvement with DLADE using dual-level embedding over the baseline system using only word embedding. We used the pairwise *t* test to examine the statistical significance of the differences in performance scores obtained from the two systems on the same test set. F1-scores of individual named entity types as well as the overall (micro-averaged) score from both systems are paired. The improvement in F1-score for DLADE as compared to our baseline is statistically significant ($p < 0.05$ and $p < 0.01$). Of all the entities, Duration showed a large improvement (11.4%) from utilizing the dual-level embedding. Duration labels are challenging to detect because they often comprise phrases that contain non-medical text and contain numbers, such as “four cycles,” “14 days,” “day 1 through 14,” “over 15 min,” and “two weeks.” They can be easily misclassified and treated as the outside or no-entity tag O.

5.4 Impact of Pre-trained Word Embedding

We demonstrated the effect of using different pre-trained word embedding in the input layer, and we compared the results from DLADE, which uses domain- and task-specific MADE1.0 word embedding trained using wiki, and Pittsburgh EHR and PubMed articles (1,352,550 word vectors) [10, 21], with two systems that use [1] general purpose GloVe Common Crawl 840B, 300 dimensional word embedding [15] (4,087,447 word vectors), and [2] the domain-specific PubMed, 200 dimensional word embedding induced from a combination of PubMed and PMC texts using the word2vec tool for biomedical data purposes [35] (2,196,016 word vectors).

Table 5 shows the F1-scores when using different pre-trained word embedding (columns 2, 3 and 4) and the percentage change in F1-scores for each type of word embedding over the others (columns 5, 6 and 7). We used the

Table 5 Percentage change in F1-scores

	F1-score with using word embedding			Percentage change in F1-scores		
	1. MADE1.0	2. GloVE	3. PubMed	4. MADE1.0 over GloVe (%)	5. MADE1.0 over PubMed (%)	6. GloVe over PubMed (%)
ADE	0.6351	0.6197	0.6055	2.48	4.88	2.34
Dose	0.8797	0.8787	0.8575	0.11	2.58	2.47
Drug	0.9042	0.9100	0.8838	− 0.63	2.31	2.96
Duration	0.7666	0.8015	0.7943	− 4.36	− 3.50	0.90
Frequency	0.8425	0.8529	0.8580	− 1.22	− 1.81	− 0.59
Indication	0.6396	0.6512	0.6429	− 1.78	− 0.52	1.28
Route	0.9239	0.9133	0.9221	1.16	0.19	− 0.96
Severity	0.8070	0.8209	0.8098	− 1.70	− 0.35	1.37
SSLIF	0.8438	0.8453	0.8454	− 0.18	− 0.19	− 0.01
Micro-averaged	0.8413	0.8451	0.8372	− 0.45	0.49	0.94

ADE adverse drug event, *SSLIF* other sign, symptom or disease

pairwise *t* test on the F1-score of the individual entity types as well as the overall (micro-averaged) score to determine if these differences are statistically significant. Our results indicate that for detecting some of the entity types, there is a minor improvement in the F1-scores when using MADE1.0 word embedding over GloVe (ADE, Dose, and Route) or PubMed (ADE, Dose, Drug, and Route). However, these improvements are not statistically significant ($p > 0.05$). Although there is a 0.49% improvement in the overall micro-averaged F1-score with MADE1.0 over PubMed, it is not statistically significant ($p > 0.05$). However, the 0.94% improvement with GloVe over PubMed is statistically significant ($p < 0.05$) for this EHR data set and ADE detection task.

6 Error Analysis of DLADE System

An error analysis was performed to understand the source of errors generated by the NER system. We inspected and evaluated instances for which the system incorrectly predicted the phrases, considering both false positive and false negative cases.

- One of the challenges as shown in Table 1 is that the entity can span across multiple words. In this case, it is critical to extract the phrase in its entirety to retain the true meaning of the phrase. For this example, our system was able to correctly extract the entire phrase “nodular sclerosing Hodgkin disease involving the mediastinum and both necks.” This contains ten words. However, the phrase was misclassified as Indication, when it actually is an *SSLIF*.
- Another challenge is the mixture of medical and non-medical text in the entity phrase. This makes it difficult to detect the entity as a whole. For instance, the phrase “inflammation of your liver or gallbladder or your pancreas” was annotated as *SSLIF*. Although our system detected the phrase correctly as *SSLIF*, it missed the last two words “your pancreas” of the phrase. This meant that our result was labeled as Other entity—O wrongly, even though it mostly was correct.
- The occurrence of medical abbreviations text is rare in the training set. Although our system was able to correctly detect certain entities that contained abbreviations, such as “stage IIA” (Severity), “HPV” (*SSLIF*), there are a few other entities with abbreviations, such as “SIL cytology” (*SSLIF*), where our system failed to recognize the phrase and categorized it as a no-entity label *O*.
- Due to the ambiguous nature of Indication, ADE, and SSLIF entity words and phrases, it is very challenging to differentiate between these two types of labels. For example, in the two sentences “the back pain [Indication] started about 10 o’clock last night” and “reports weight gain [*ADE*] and increased [*ADE*] appetite from corticosteroid therapy,” our system misclassified the Indication and *ADE* labels as *SSLIF*.

7 Discussion

In this paper, we report our experience with the MADE1.0 competition and describe our system, which was ranked first in the NER task. We study the problem of detecting ADE-related terms and phrases from EHRs. Unlike other research domains such as computer vision where billions of labeled images are made publicly available for research purposes, making large-scale labeled medical corpus publicly available is an open challenge as it is a human resource–intensive task and it involves many legal issues. We appreciate the effort of the MADE organizers who provided annotated EHRs. However, the size of the corpus is still considerably small. The generality of our method could not be validated even with the great results we have shown in the experiments.

For this competition, we used the same methodology for all entity types. However, the challenge of detecting each individual entity type varies. For example, the ability to capture morphological features is important to the entity types if they often consist of special representations as compared to common words, whereas the ability to capture the context information is crucial to differentiate *Indication* and *ADE* as they may share the same vocabulary, but are expressed differently in the text narrative.

Overall, our system achieved excellent detection accuracy, with a micro-averaged precision of 0.8373, recall of 0.8454 and F1-score of 0.8413. However, the detection accuracy among the nine individual entity types varied, with some entity types achieving better F1-scores, such as Route (0.92), Drug (0.90), Dose (0.88), SSLIF (0.84), Frequency (0.84), Severity (0.81), and Duration (0.77), over other entity types, such as ADE (0.64) and Indication (0.64). Given that ADE and related information detection from EHR is a challenging task, our system showed an incremental improvement in the scores compared to the benchmark studies [10, 21]. Yet ADE and Indication have proved to be the most challenging of the entity types to detect, with a lower recall (0.56 and 0.62, respectively). These challenging entity types might require customized models that are able to tackle the issues with ambiguity that is often encountered while detecting these entities.

8 Conclusion

We have shown that the integration of two widely used sequence labeling techniques that complement each other along with dual-level embedding (character level and word level) to represent words in the input layer results in a deep learning architecture that achieves excellent information extraction accuracy for EHR notes. Our system was ranked first in the MADE1.0 competition for the NER task. Additional work must be done to improve the accuracy in detecting the challenging entity types such as ADE and Indication. In the future, we will further analyze the results for these entity types and design customized models to improve the detection performance of each individual entity type as well as the overall performance for all entity types.

Acknowledgments We are grateful to Dr. Marni Hall (former Sr. Program Director), Suranjan De (Deputy Director), Sanjay K. Sahoo and Thang La, Regulatory Science, Office of Surveillance and Epidemiology (OSE), US Food and Drug Administration (FDA) for introducing us to pharmacovigilance in general and the ADE detection problem in particular. We also thank Worcester Polytechnic Institute (WPI) Data Science Research Group (DSRG) members for their valuable feedback.

Compliance with Ethical Standards

Conflict of interest Susmitha Wunnava, Xiao Qin, Tabassum Kakar, Cansu Sen, Elke A. Rundensteiner and Xiangnan Kong have no conflicts of interest that are directly relevant to the content of this study.

Funding Susmitha Wunnava is thankful to the Seeds of STEM and Institute of Education Sciences, US Department of Education for supporting her PhD studies via the Grant R305A150571. Xiao and Tabassum are grateful to Oak Ridge Associated Universities (ORAU) for granting them an ORISE Fellowship to conduct research with the US Food and Drug Administration.

References

1. Donaldson MS, Corrigan JM, Kohn LT, editors. To err is human: building a safer health system. Washington DC: National Academies Press; 2000.
2. Wunnava S, Qin X, Kakar T, Kong X, Rundensteiner EA, Sahoo SK, et al. One size does not fit all: an ensemble approach towards information extraction from adverse drug event narratives. In: Proceedings of HEALTHINF; 2018. pp 176–188.
3. Deleger L, Grouin C, Zweigenbaum P. Extracting medical information from narrative patient records: the case of medication-related information. J Am Med Inform Assoc. 2010;17(5):555–8.
4. Xu H, Stenner SP, Doan S, Johnson KB, Waitman LR, Denny JC. MedEx: a medication information extraction system for clinical narratives. J Am Med Inform Assoc. 2010;17:19–24.
5. Sampathkumar H, Xw Chen, Luo B. Mining adverse drug reactions from online healthcare forums using hidden Markov model. BMC Med Inf Decis Mak. 2014;14:91.
6. Ramesh BP, Belknap SM, Li Z, Frid N, West DP, Yu H. Automatically recognizing medication and adverse event information from food and drug administration's adverse event reporting system narratives. JMIR. 2014;8:2.
7. Lipton ZC. A Critical review of recurrent neural networks for sequence learning. CoRR. 2015; abs/1506.00019.
8. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9:1735–80.
9. Lafferty J, McCallum A, Pereira FCN. Conditional random fields: probabilistic models for segmenting and labeling sequence data. 2001.
10. Jagannatha AN, Yu H. Structured prediction models for RNN based sequence labeling in clinical text. In: Proceedings of the conference on empirical methods in natural language processing. In: Conference on empirical methods in natural language Processing; 2016.
11. Tutubalina E, Nikolenko S. Combination of deep recurrent neural networks and conditional random fields for extracting adverse drug reactions from user reviews. J Healthcare Eng; 2017;2017: Article ID 945134, 9.
12. Huang Z, Xu W, Yu K. Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991. 2015.
13. Dubois S, Romano N. Learning effective embeddings from medical notes.
14. Choi Y, Chiu CYI, Sontag D. Learning low-dimensional representations of medical concepts. In: AMIA summits on translational science proceedings. 2016.

15. Pennington J, Socher R, Manning C. Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP); 2014. pp 1532–1543.
16. Wunnava S, Qin X, Kakar T, Rundensteiner EA, Kong X. Bidirectional LSTM-CRF for adverse drug event tagging in electronic health records. In Liu F, Jagannatha A, Yu H, editors. In: Proceedings of the 1st international workshop on medication and adverse drug event detection, volume 90 of Proceedings of machine learning research; 2018 May 4. pp 48–56.
17. Comeau DC, Islamaj Dogan R, Ciccarese P, Cohen KB, Krallinger M, Leitner F, et al. BioC: a minimalist approach to interoperability for biomedical text processing. Database. 2013; 2013.
18. Bird S, Loper E. NLTK: the natural language toolkit. In: Proceedings of the ACL 2004 on Interactive poster and demonstration sessions; 2004. p 31.
19. Manning C, Surdeanu M, Bauer J, Finkel J, Bethard S, McClosky D. The Stanford CoreNLP natural language processing toolkit. In: Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations; 2014. pp 55–60.
20. Savova GK, Masanz JJ, Ogren PV, Zheng J, Sohn S, Kipper-Schuler KC, et al. Mayo clinical text analysis and knowledge extraction system (cTAKES): architecture, component evaluation and applications. J Am Med Inform Assoc. 2010;17:507–13.
21. Jagannatha AN, Yu H. Bidirectional RNN for medical event detection in electronic health records. In: Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting; 2016. p 473.
22. Aronson AR. Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. In: Proceedings of the AMIA Symposium; 2001.
23. Bird S, Klein E, Loper E. Natural language processing with python: O'Reilly; 2009.
24. Ramshaw LA, Marcus MP. Text chunking using transformation-based learning. In Natural language processing using very large corpora. Springer; 1999. Pp 157–176.
25. Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P. Natural language processing (almost) from scratch. J Mach Learn Res. 2011;12:2493–537.
26. Santos CD, Zadrozny B. Learning character-level representations for part-of-speech tagging. In: Proceedings of the 31st international conference on machine learning (ICML-14); 2014. pp 1818-1826.
27. Bengio P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw. 1994;5:157–66.
28. Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks. In: International conference on machine learning; 2013. pp 1310–1318.
29. Gers FA, Schmidhuber J, Cummins F. Learning to forget: continual prediction with LSTM. IET. 1999.
30. Schuster M, Paliwal KK. Bidirectional recurrent neural networks. IEEE Trans Signal Process. 1997;45:2673–81.
31. Ma X, Hovy EH. End-to-end Sequence Labeling via bi-directional LSTM-CNNs-CRF. In Proceedings of the 54th annual meeting of the association for computational linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers; 2016.
32. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res. 2014;15:1929–58.
33. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980. 2014.
34. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, et al. TensorFlow: a system for large-scale machine learning. OSDI. 2016;16:265–83.
35. Pyysalo S, Ginter F, Moen H, Salakoski T, Ananiadou S. Distributional semantics resources for biomedical text processing. In Proceedings of the 5th international symposium on languages in biology and medicine. Tokyo, Japan; 2013. pp 39–43