Journal of Biomedical Informatics 76 (2017) 102–109

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Recurrent neural networks with specialized word embeddings for health-domain named-entity recognition

Iñigo Jauregi Unanue[a,b,*], Ehsan Zare Borzeshi[b], Massimo Piccardi[a]

[a] University of Technology Sydney (UTS), Australia
[b] Capital Markets Cooperative Research Centre (CMCRC), Australia

ARTICLE INFO

Keywords:
Neural networks (computer) [MeSH]
Machine learning [MeSH]
Artificial intelligence [MeSH]
Clinical concept extraction
Drug name recognition

ABSTRACT

Background: Previous state-of-the-art systems on Drug Name Recognition (DNR) and Clinical Concept Extraction (CCE) have focused on a combination of text "feature engineering" and conventional machine learning algorithms such as conditional random fields and support vector machines. However, developing good features is inherently heavily time-consuming. Conversely, more modern machine learning approaches such as recurrent neural networks (RNNs) have proved capable of automatically learning effective features from either random assignments or automated word "embeddings".

Objectives: (i) To create a highly accurate DNR and CCE system that avoids conventional, time-consuming feature engineering. (ii) To create richer, more specialized word embeddings by using health domain datasets such as MIMIC-III. (iii) To evaluate our systems over three contemporary datasets.

Methods: Two deep learning methods, namely the Bidirectional LSTM and the Bidirectional LSTM-CRF, are evaluated. A CRF model is set as the baseline to compare the deep learning systems to a traditional machine learning approach. The same features are used for all the models.

Results: We have obtained the best results with the Bidirectional LSTM-CRF model, which has outperformed all previously proposed systems. The specialized embeddings have helped to cover unusual words in *DrugBank* and *MedLine*, but not in the *i2b2/VA* dataset.

Conclusions: We present a state-of-the-art system for DNR and CCE. Automated word embeddings has allowed us to avoid costly feature engineering and achieve higher accuracy. Nevertheless, the embeddings need to be re-trained over datasets that are adequate for the domain, in order to adequately cover the domain-specific vocabulary.

1. Introduction

In recent years, the amount of digital information generated from all sectors of society has increased rapidly, and as a result, agriculture, industry, small businesses and, of course, healthcare, are becoming more efficient and productive thanks to the insights obtained from the "Big Data". However, in order to deal effectively with such large data, there is an ongoing need for novel, scalable and more accurate analytic tools.

In the healthcare system, patients' medical records represent a big data source. Even though the records contain very useful information about the patients, in most cases the information consists of unstructured text such as, among others, doctors' notes, medical observations made by various physicians, and descriptions of the recommended treatments. This type of data cannot be analyzed using common statistical tools; rather, they need to be approached by Natural Language Processing (NLP) techniques. In this paper, we focus on a well-known task in NLP, namely Named-Entity Recognition (NER). The goal of NER is to automatically find "named entities" in text and classify them into predefined categories such as people, locations, companies, time expressions etc. In the case of specialized domains, NER systems focus on text with specific dictionaries and topics, together with dedicated sets of named-entities. In the health domain, the two most important NER tasks are Clinical Concept Extraction (CCE) and Drug Name Recognition (DNR). The former aims to identify mentions of clinical concepts in patients' records to help improve the organization and management of healthcare services. Named entities in CCE can include test names, treatments, problems related to individual patients, and so forth. The latter seeks to find drug mentions in unstructured biomedical texts to match drug names with their effects and discover drug-drug interactions (DDIs). DNR is a key step of pharmacovigilance (PV) which is concerned with the detection and understanding of

* Corresponding author at: University of Technology Sydney (UTS), Australia.
E-mail address: ijauregi@cmcrc.com (I. Jauregi Unanue).

https://doi.org/10.1016/j.jbi.2017.11.007
Received 9 June 2017; Received in revised form 5 September 2017; Accepted 11 November 2017
Available online 13 November 2017
1532-0464/

Sentence	the	effects	of	chronic	phenyotin	or	carbamazepine	therapy
Entity class	O	O	O	O	B-drug	O	B-drug	O

a) DNR example

Sentence	his	lateral	percutaneous	drains	had	been	pulled	out
Entity class	B-treatment	I-treatment	I-treatment	I-treatment	O	O	O	O

b) CCE example

Fig. 1. (a) DNR and (b) CCE tasks examples, where 'B' (beginning) specifies the start of a named entity, 'I' (inside) specifies that the word is part of the same named entity, and 'O' (outside) specifies that the word is not part of any predefined class.

adverse effects of drugs and other drug-related problems. Fig. 1 shows examples of both tasks.

NER is a challenging learning problem because in most domains the training datasets are scarce, preventing a "brute-force" approach by exhaustive dictionaries. Consequently, many systems rely on handcrafted rules and language-specific knowledge to solve this task. To give a simple example of such rules, if the word begins with a capital letter in the middle of the sentence, it can be assumed to be a named entity in most cases. Nevertheless, these approaches are time-costly to develop, depend considerably on the language and the domain, are ineffective in the presence of informal sentences and abbreviations and, although they usually achieve high precision, suffer from low recall (i.e., they miss many entities). Conversely, machine learning (ML) approaches overcome all these limitations as they are intrinsically robust to variations. Current state-of-the-art ML methods follow a two-step process: (1) feature engineering and (2) automated classification [1–4]. The first step represents the text by numeric vectors using domain-specific knowledge. The second step refers to the task of classifying each word into a different named-entity class, with popular choices for the classifier being the linear-chain Conditional Random Fields (CRF), Structural Support Vector Machines (S-SVM) and maximum-entropy classifiers. The drawback of this approach is that feature engineering can be often as time-consuming as the manual design of rules.

In recent years, the advent of deep learning has contributed to significantly overcome this problem [5–7]. The Long Short-Term Memory (LSTM) and its variants (e.g., the Bidirectional LSTM), which are a specific type of Recurrent Neural Networks (RNNs), have reported very promising results [5]. In these models, words only need to be assigned to random vectors, and during training the neural network is able to automatically learn improved representations for them, completely bypassing feature engineering. In order to further increase the performance of these systems, the input vectors can alternatively be assigned with general-purpose word embeddings learned with GloVe or Word2vec [8,9]. The aim of general-purpose word embeddings is to map every word in a dictionary to a numerical vector (the embedding) so that the distance between the vectors somehow reflects the semantic difference between the words. For example, 'cat' and 'dog' should be closer in the vector space than 'cat' and 'car'. The common principle behind embedding approaches is that the meaning of a word is conveyed by the words it is used with (its surrounding words, or context). Therefore, the training of the word embeddings only requires large, general-purpose text corpora such as Wikipedia (400 K unique words) or Common Crawl (2.2 M unique words), without the need for any manual annotation. However, drug and clinical concept recognition are very domain-specific tasks, and many words might not appear in general-domain datasets. In order to assign word embeddings to these specialized words, the embedding algorithms need to be retrained using medical domain resources such as the MIMIC-III corpora [10]. As well as semantic word embeddings, *character-level* embeddings of words can also be automatically learned. Such embeddings can capture typical prefixes and suffixes, providing the classifiers with richer representations of the words [5].

Preliminary results for the work presented in this paper have obtained very promising accuracy in DNR and CCE tasks using neural networks. Chalapathy et al. [11] presented a DNR system that uses a Bidirectional LSTM-CRF architecture with random assignments of the input word vectors at the EMNLP 2016 Health Text Mining and Information Analysis workshop. The reported results were very close to the system that ranked first in the *SemEval-2013 Task 9.1*. In Chalapathy et al. [12], the authors leveraged the same architecture for CCE at the Clinical NLP 2016 workshop, this time using pre-trained word embeddings from GloVe, and the results outperformed previous systems over the *i2b2/VA* dataset. In this paper, we extend the previous research by training the deep networks with more complex and specialized word embeddings. Moreover, we explore the impact of augmenting the word embeddings with conventional feature engineering. As methods, we compare contemporary recurrent neural networks such as the Bidirectional LSTM and the Bidirectional LSTM-CRF against a conventional ML baseline (a CRF). We report state-of-the-art results in both DNR and CCE.

2. Related work

Most of the research carried out in domain-specific NER has combined supervised and semi-supervised ML models with text feature engineering. For example, the WBI-NER system that ranked first in the SemEval-2013 Task 9.1 (Recognition and classification of pharmacological substances, DNR) [3], is based on a linear-chain CRF with specialized features. Other similar systems for DNR [2,13] use various general- and domain-specific features. In CCE, the same approach (feature engineering + conventional ML classifier) has achieved the best results [4,14].

In the recent years, there has been an increase in the use of deep neural networks for a variety of NLP tasks, including NER [5–7]. Pre-trained word embeddings [8,9,15] have been used in traditional ML methods [16,17] and in neural networks, where Deconourt et al. [18] has achieved better performance than previously published systems in de-identification of patient notes. Cocos et al. [19] have used the Bidirectional LSTM model for labelling Adverse Drug Reactions in pharmacovigilance. Xie et al. [20] have used a similar model for studying the adverse effects of e-cigarettes. Wei et al. [21] have combined the output of a Bidirectional LSTM and a CRF as input to an SVM classifier for disease name recognition. A possible drawback of this approach is that the overall prediction is not structured and may miss on useful correlation between the output variables.

In a work that is more related to ours, Jaganatha and Yu [22] have employed a Bidirectional LSTM-CRF to label named entities from electronic health records of cancer patients. Their model differs in the CRF output module where the pairwise potentials are modelled using a Convolutional Neural Network (CNN) rather than the usual transition matrix. Gridach [23] has also used the Bidirectional LSTM-CRF for named-entity recognition in the biomedical domain.

The main difference and contribution of the proposed approach is that it leverages specialized health-domain embeddings created from a

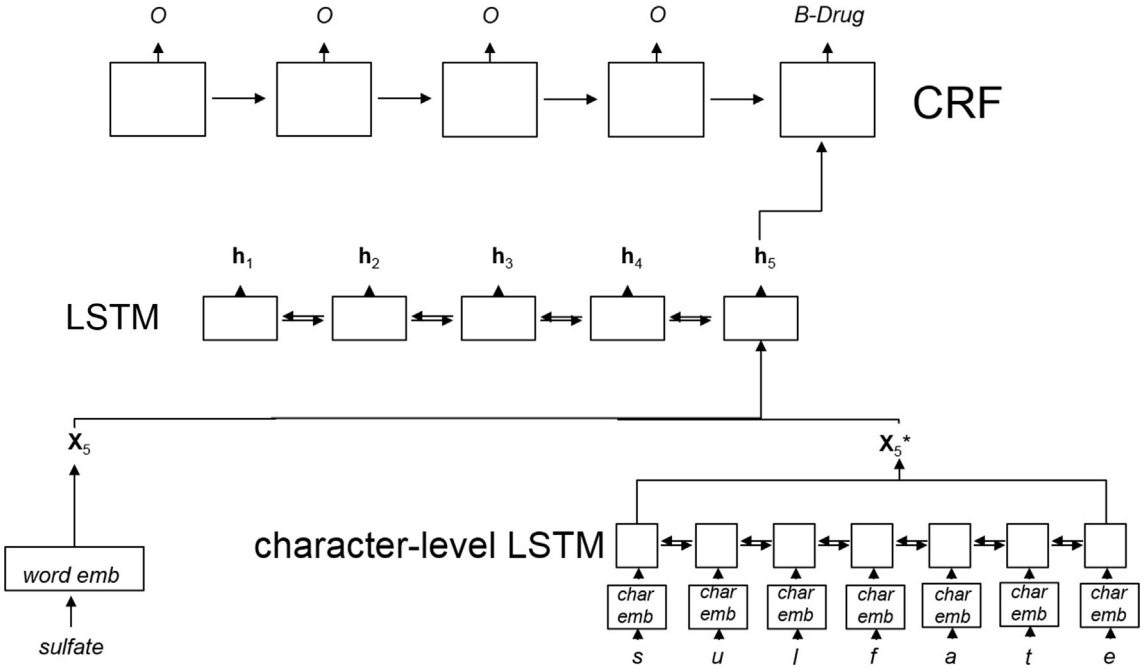

Fig. 2. The Bidirectional LSTM-CRF with word-level and character-level word embeddings. In the example, word 'sulfate' is assumed to be the 5th word in a sentence and its only entity;'x_5' represents its word-level embedding (a single embedding for the whole word); 'x_5*' represents its character-level embedding, formed from the concatenation of the last hidden state of the forward and backward passes of a character-level Bidirectional LSTM; 'h_{1-5}' are the hidden states of the main Bidirectional LSTM which become the inputs into a final CRF; eventually, the CRF provides the pr labeling.

structured database. In the experiments, these embeddings have been used jointly with general-domain embeddings and they have proved able to improve the accuracy in several cases. In addition, our work evaluates the use of hand-crafted features in the system [24]. This aims to provide a comprehensive feature comparison for health-domain named-entity recognition based on LSTM models.

3. Methods

In this section we provide a description of the main methods employed. First, we describe the conditional random field (CRF), a traditional machine learning approach for the classification of sequences, which is used as a baseline in the experiments. This baseline is compared with two variants of a contemporary recurrent neural network, which are known as Bidirectional LSTM and Bidirectional LSTM-CRF, respectively.

3.1. CRF

A CRF model is a well-known machine learning approach that has been widely used in NER [25]. It predicts sequences of labels ($\boldsymbol{y}$) from sequences of measurements ($\boldsymbol{x}$) taking into account the sequentiality of the data. A CRF model, $p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})$, is given in Eq. (1) below, where $\boldsymbol{w}$ notes the model's parameters, $\Psi(\boldsymbol{x},\boldsymbol{y})$ is the chosen feature vector and $Z(\boldsymbol{w},\boldsymbol{x})$ is the cumulative sum of $p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})$ over all the possible $\boldsymbol{y}$:

$$p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w}) = \frac{\exp(\boldsymbol{w}^T\Psi(\boldsymbol{x},\boldsymbol{y}))}{Z(\boldsymbol{w},\boldsymbol{x})} \quad (1)$$

The parameters of this model are typically learned from a training set, $(\boldsymbol{Y},\boldsymbol{X}) = \{\boldsymbol{x}_i,\boldsymbol{y}_i\}, i = 1...N$, with conditional maximum likelihood as in:

$$\boldsymbol{w} = arg\ max_{\boldsymbol{w}} p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w}) \quad (2)$$

Once the model has been trained, the prediction of a CRF is the sequence of labels maximizing the model for the given the input sequence and the learned parameters:

$$\boldsymbol{y}^* = arg\ max_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w}) \quad (3)$$

The labels are typically predicted using a Viterbi-style algorithm which provides the optimal prediction for the measurement sequence as a whole. The model is trained by maximizing the conditional likelihood, or cross-entropy, over a given training set. For its implementation, we have used the HCRF library [26]. The features used as input are described in Section 4. In the experiments, we use the CRF as a useful baseline for performance comparison with the proposed neural networks. Note that a CRF model is also used as the output layer in the Bidirectional LSTM-CRF as explained in the next section.

3.2. Bidirectional LSTM and bidirectional LSTM-CRF

RNNs are a type of neural network architecture in which connections between units form a directed cycle, creating an internal state and achieving dynamic temporal behavior. Thanks to their internal memory, RNNs can process a sequence of vectors ($\boldsymbol{x}_1, \boldsymbol{x}_2,\ldots, \boldsymbol{x}_n$) as input and produce another sequence ($\boldsymbol{h}_1, \boldsymbol{h}_2,\ldots, \boldsymbol{h}_n$) as output that contains some extent of sequential information about every vector in the input. However, these architectures in practice fail to learn long-term dependencies in the sequences as they tend to be biased by the most recent vectors [27]. The Long Short-Term Memory (LSTM) was therefore designed to overcome this issue by incorporating a gated memory-cell that has been shown to capture long-term dependencies [28]. Eq. (4) shows the implementations of the different gates in the LSTM [5], where $\boldsymbol{i}_t$ is the "input" gate, $\boldsymbol{c}_t$ is the "cell" gate, $\boldsymbol{o}_t$ is the "output" gate, $\boldsymbol{W}$ are the weights of the network, $\boldsymbol{b}$ are the biases, σ is the element-wise sigmoid function, and $\odot$ is the element-wise product. The bidirectional LSTM (B-LSTM) is just a variation, in which both the left-to-right ($\overrightarrow{\boldsymbol{h}_t}$) and the right-to-left ($\overleftarrow{\boldsymbol{h}_t}$) representations of the input sentence are generated, and then concatenated $\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}_t};\overleftarrow{\boldsymbol{h}_t}]$ in order to obtain the final representation.

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{ci}\boldsymbol{c}_{t-1} + \boldsymbol{b}_i)$$
$$\boldsymbol{c}_t = (1-\boldsymbol{i}_t) \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tanh(\boldsymbol{W}_{xc}\boldsymbol{x}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c)]$$
$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{co}\boldsymbol{c}_t + \boldsymbol{b}_o)$$
$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) \quad (4)$$

When applying the LSTM in NER, the words in the input sentence are first mapped to numerical vectors. These vectors can be random valued, a pre-trained word embedding, domain-specific word features or any combination of them. For each vector, the output of the network are the posterior probabilities of each named-entity class. An improvement of these networks has been presented by Lample et al. [5] using a CRF as a final output layer. This final layer provides the system with the ability to perform joint decoding of the input sequence in a Viterbi-style manner. The resulting network is known as the Bidirectional LSTM-CRF (B-LSTM-CRF). We test the LSTM models with the same features used for the CRF in order to establish the fairest-possible comparison. The features are described in detail in Section 4. Fig. 2 shows a descriptive diagram of the Bidirectional LSTM-CRF.

4. Word features

As mentioned above, neural networks can learn meaningful representations from random initializations of word embeddings. However, it has been proved that pre-trained word embeddings can improve the performance of the network [5,11,18,24]. In this section, we present the pre-trained embeddings employed in lieu of the random assignments.

4.1. Specialized word embeddings

A word embedding maps a word to a numerical vector in a vector space, where semantically-similar words are expected to be assigned similar vectors. To perform this mapping, we have used a well-known algorithm called GloVe [8]. This algorithm learns word embeddings by looking at the co-occurrences of the word in the training data, assuming that a word's meaning is mostly defined by its context and, therefore, words having similar contexts should have similar embeddings. GloVe can be trained from large, general-purpose datasets such as Wikipedia, Gigaword5 or Common Crawl without the need for any manual supervision. In this work, we have experimented with different general-purpose, pre-trained word embeddings from the official GloVe website [29] and noticed that the embeddings trained with Common Crawl (cc) (2.2 M unique words) were giving the best results. We have employed these embeddings on their own, and also concatenated with the MIMIC-III embeddings (cc/mimic). By default, the code always initializes the word embedding of each unique word in the dictionary with a unique random vector. In alternative, we replace the random initialization with a pre-trained embedding. However, although such datasets generate good embeddings in many cases, for domain-specific tasks such as DNR and CCE they can suffer from some lack of vocabulary. As a matter of fact, in health corpora it is common to find very technical and unusual words which are specific to the health domain. If GloVe is trained only with general-purpose datasets, it is likely that such words will be missing and will still have to be assigned with random vectors.

In order to solve this problem, we have generated a new word embedding by re-training GloVe with a large health domain dataset called MIMIC-III [10]. This dataset contains records of 53,423 distinct hospital admissions of adults to an intensive care unit between 2001 and 2012. The data, structured in 26 tables, include information such as vital signs, observations of care providers, diagnostic codes etc. We expect such a dataset to contain many of the technical words from the health domain that may not appear in general-domain datasets, and as the size of MIMIC-III is sufficiently large, we should be able to extract meaningful vector representations for these words. As a first step, we have selected a subset of the tables and columns, and generated a new dataset where each selected cell together with the title of the corresponding column form a pseudo-sentence. As the next step, we have used this dataset to re-train GloVe, and concatenated these specialized word embeddings with the others to create vectors that contain information from both approaches. Obviously, there are words that appear in the general dataset, but not in MIMIC-III, and the vice versa. In such cases, the corresponding embedding is still assigned randomly. If a word does not appear in either dataset, we assign its whole embedding randomly. In all cases, the embeddings are updated during training by the backpropagation step.

4.2. Character-level embeddings

Following Lample et al. [5] we also add character-level embeddings of the words. Such embeddings reflect the actual sequence of characters of a word and have proven to be useful for specific-domain tasks and morphologically-rich languages. Typically, they contribute to catching prefixes and suffixes which are frequent in the domain, and correctly classifying the corresponding words. As an example, a word ending in "cycline" is very likely a drug name, and a character-level embedding could help classify it correctly even if the word was not present in the training vocabulary. All the characters are initialized with a random embedding, and then the embeddings are passed character-by-character to a dedicated LSTM in both forward and backward order. The final outputs in the respective directions promise to be useful encodings of the ending and the beginning of the word. These character-level embeddings are integral part of the LSTM architecture and are not available in the CRF or other models. The character embeddings, too, are updated during training with backpropagation.

4.3. Feature augmentation

Conventional machine learning approaches for NER usually have a feature engineering step. Lee et al. [24] have shown that adding hand-crafted features to a neural network can contribute to increase the recall. In our work, we try this approach with features similar to those used by Lee et al. [24] Fig. 3 shows the list of features used. The distinct values of each feature are encoded onto short random vectors, for a total dimension of 146-D. During training, these encodings are updated as part of the backpropagation step.

Feature Types	Features
Morphological	Is all lowercase, is all uppercase, has first letter capitalized, has a letter in the middle capitalized, ends with s, contains digits, is numeric, is alphabetic, is alphanumeric, is a stop word
Semantic	POS tagging, lemma, UMLS concept extracted with MetaMap
Clustering	Index of cluster to which the word embedding belongs. The embeddings clustered (K=20) are the concatenation of Common Crawl and MIMIC-III embeddings (dim=600).

Fig. 3. Description of the hand-crafted features.

Table 1
Statistics of the training and test datasets used in the experiments.

	Training set	Test set
(a) i2b2/VA		
Documents	170	256
Sentences	16,315	27,626
problem	7073	12,592
test	4608	9225
treatment	4844	9344

	DrugBank		MedLine	
	Training set	Test set	Training set	Test set
(b) SemEval-2013 Task 9.1				
Documents	730	54	175	58
Sentences	6577	145	1627	520
drug_n	124	6	520	115
group	3832	65	234	90
brand	1770	53	36	6
drug	9715	180	1574	171

5. Results

5.1. Datasets

Hereafter, we evaluate the models on three datasets in the health domain. The first is the *2010 i2b2/VA IRB Revision* (we refer to it as *i2b2/VA* for short in the following) and is used for evaluating CCE. This dataset is a reduced version of the original 2010 i2b2/VA dataset that is no longer distributed due to restrictions introduced by the Institutional Review Board (IRB) in 2011 [30]. The other two datasets are *DrugBank* and *MedLine*, both part of the *SemEval-2013 Task 9.1* for DNR [31]. Table 1a and b describes the basic statistics of these datasets. For the experiments, we have used the official training and test splits released with the distributions.

5.2. Evaluation metrics

We report the performance of the model in terms of the F1 score. The F1 score is a very relevant measure as it considers both the precision and the recall, computing a weighted average of them. If we note as TP the number of true positives, FP the false positives and FN the false negatives, we have:

- $precision = \frac{TP}{TP + FP}$
- $recall = \frac{TP}{TP + FN}$
- $F1 = \frac{2\ precision \cdot recall}{precision + recall}$

However, it must be remarked that there are different ways of computing the precision and the recall, depending on what we consider as a correct or incorrect prediction [32]. In this work, we employ the "strict" evaluation method, where both the entity class and its exact boundaries are expected to be correct. We have used the B-I-O tagging standard to annotate the text at word level. In detail, 'B' means the beginning (first word) of a named entity; 'I' stands for 'inside', meaning that the word is part of the same entity (for multi-word entities; e.g., "albuterol sulfate"); and 'O' stands for 'outside', meaning that the word is not part of any named entities. Therefore, a valid annotation of a named entity always begins with a 'B'. An example is shown in Fig. 4. All the models used in this paper have been trained to predict explicit 'B' and 'I' labels for each entity class. The evaluation includes a pre-processing step that converts an 'I' prediction to a 'B' if it follows directly an 'O' prediction, thus making all predicted entities valid. An entity is considered as correctly predicted only if all its 'B' and 'I' labels and all its classes are predicted correctly. In the example of Fig. 4 the prediction will be counted as a true positive only if all the four words "recently diagnosed abdominal carcinomatosis" are tagged as a single entity of the problem class. Every differing 'B' prediction will instead be counted as a false positive. The evaluation protocol explicitly counts only the true positives and the false positives, and derives the false negatives as (number of true entities – true positives).

5.3. Training and hyper-parameters

For an unbiased evaluation, all the trained models have been tested blindly on unseen test data. In order to facilitate replication of the empirical results, we have used a publicly-available library for the implementation of the neural networks (i.e. the Theano neural network toolkit [33]) and we release our code [34]. To operate, any machine learning model requires both a set of parameters, which are learned automatically during training, and some "hyper-parameters", which have to be selected manually. Therefore, we have divided the training set of each dataset into two parts: a training set for learning the parameters (70%), and a validation set (30%) for selecting the best hyper-parameters [35]. The hyper-parameters of the LSTM include the number of hidden nodes (for both LSTM versions), $(H_w,H_c) \in \{25,50,100\}$; the word embedding dimension, $dw \in \{50,100,200,300,600\}$; and the character embedding dimension, $d_c \in \{25,50,100\}$. Additional hyper-parameters include the learning rate and the drop-out rate, which were left to their default values of [0.01] and [0.5] respectively [36]. All weights in the network, feature encodings and the words that do not have a pre-trained word embedding have been initialized randomly from the uniform distribution within range $[-1, 1]$, and updated during training with backpropagation. The number of training "epochs" (i.e., iterations) was set to 100, selecting the epoch that obtained the best results on the validation set. The best model from the validation set was finally tested on the unseen, independent test set without any further tuning, and the corresponding accuracy reported in the tables. Table 2 shows all the hyper-parameters used for the experiments reported in the Results section.

5.4. Results

Table 3a and b shows the results of the proposed models and the state-of-the-art systems on the CCE task (*i2b2/VA* dataset) and DNR task (*DrugBank and MedLine* datasets), respectively. In the following subsections, we discuss the results obtained for each task.

5.4.1. CCE results over the i2b2/VA dataset

On the *i2b2/VA* dataset (Table 3a), the Bidirectional LSTM-CRF (B-LSTM-CRF) with Common Crawl embeddings (cc) and character-level embeddings (char) as features has obtained the best results (83.35% F1 score). The model has outperformed all systems from the literature (top quadrant of Table 3a) which are all based on conventional domain-specific feature engineering. It is important to note that deBruijin et al. [4] had reported a higher accuracy on 2010 i2b2/VA (85.23% F1 score), but their model was trained and tested on the original version of the dataset which is no longer available due to the restrictions introduced by the Institutional Review Board. As for what specialized embeddings are concerned, Table 4 shows that the general-domain dataset Common Crawl already contains almost all the words in the dataset. Therefore, adding the MIMIC-III embeddings (mimic) does not extend the vocabulary, and therefore it brings no improvement. On the other hand, the B-LSTM has improved by 0.3 pp with the cc/mimic embeddings. Even though the mimic embeddings do not cover significant extra vocabulary, they may have enriched the feature space. Conversely, the cc/mimic embeddings have provided no improvements with the B-LSTM-CRF. For this, we need to take into account that the B-LSTM-CRF already has a high score (83.35% F1-score). Consequently, it may be more difficult to improve its results. Conversely, using

a)	*Gentleman*	*with*	*recently*	*diagnosed*	*abdominal*	*Carcinomatosis*
	O	*O*	*O*	*O*	*O*	*B-problem*
b)	*Gentleman*	*with*	*recently*	*diagnosed*	*abdominal*	*Carcinomatosis*
	O	*O*	*B-problem*	*I-problem*	*I-problem*	*I-problem*

Fig. 4. (a) An example of an incorrect tagging in the "strict" evaluation method. (b) An example of a correct tagging in the "strict" evaluation method.

Table 2
The hyper-parameters used in the experiments.

Hyper-parameter	Value
Word embedding dim (d_w)	300 (cc)/600 (cc + mimic)
Word LSTM hidden layer dim (H_w)	100
Char embedding dim (d_c)	25
Char LSTM hidden layer dim (H_c)	25
Dropout	0.5
Optimization	Stochastic Gradient Descend
Learning rate	0.01
Concatenated hand-crafted features dim	146

conventional feature engineering has led to lower accuracy (77.81% F1-score). Eventually, concatenating both the features and the pre-trained embeddings showed no improvement over the best model. Table 3a also shows the importance of using a final CRF layer in the B-LSTM-CRF, given that the B-LSTM alone was only able to achieve a 77.59% F1 score. At its turn, the CRF baseline has only obtained a 64.09% F1 score in its best configuration, lower than any version of the LSTM.

5.4.2. DNR results over the DrugBank and MedLine datasets

In the DNR task (Table 3b), the proposed B-LSTM-CRF with the concatenated word embeddings (cc/mimic) and the character-level embeddings (char) has improved over all the previous approaches on both *DrugBank* (88.38% F1 score) and *MedLine* (60.66% F1 score). Table 4 shows that only 49% of the words in the datasets have been found in the cc embeddings. However, when the concatenated embeddings (cc/mimic) are used, the percentage of found words has increased to 67% for *DrugBank* and 61% for *MedLine*, leading to better results in the classification task. Words that appear in the MIMIC-III dataset but are not contained in Common Crawl are typically very technical and domain-specific, such as drug names or treatments; examples include: *pentostatin*, *sitagliptin*, *hydrobromide*, *organophosphate*, *pyhisiological* and *methimazole*. In total, 1189 extra words have been mapped in *DrugBank* and 716 in *MedLine* thanks to the use of MIMIC-III. However, the B-LSTM has only obtained an accuracy improvement on the *MedLine* dataset, but not on *DrugBank*. This can be explained by the fact that the accuracy of the B-LSTM on *MedLine* is very low (44.33%) and, therefore, easy to improve. Instead, on *DrugBank* the accuracy of the B-LSTM is already very high (84.35% F1-score) and thus difficult to improve. With the B-LSTM-CRF, results with extra vocabulary covered by the cc/mimic embeddings have improved with both datasets.

As for what concerns the hand-crafted features, their use has led to higher accuracy than with the Common Crawl embeddings on the *DrugBank* dataset in two cases. However, the concatenation of the features and the pre-trained embeddings has not improved the best results. As in the CCE task, the B-LSTM-CRF model has proved better than the B-LSTM alone on both *DrugBank* (88.38% vs 84.35% F1-score) and *MedLine* (60.66% vs 45.92% F1-score.) Finally, we can see that the use of the character-level embeddings has led to higher relative improvements for *DrugBank* than for the other two datasets. A plausible explanation for this is that this dataset contains more words with distinctive prefixes and suffixes which are more effectively captured by the character-level embeddings.

In general, the CRF has significantly underperformed compared to the neural networks. We speculate that this model may require more extensive feature engineering to achieve a comparable performance, or

Table 3
Comparison of the results between the different RNN models and the state-of-the-art systems over the CNE and DNR tasks.

Model	i2b2/VA F1 score (%)
(a) CCE results over the i2b2/VA dataset	
Binarized Neural Embedding CRF [17]	82.80
CliNER [14]	80.00
Truecasing CRFSuite [37]	75.86
CRF + (random)	11.27
CRF + (features)	25.53
CRF + (cc)	53.72
CRF + (cc/mimic)	58.28
CRF + (cc/mimic) + (features)	64.09
B-LSTM + (random)	65.43
B-LSTM + (random) + (features)	69.42
B-LSTM + (cc)	75.17
B-LSTM + (cc) + (char)	76.79
B-LSTM + (cc/mimic) + (char)	77.19
B-LSTM + (cc/mimic) + (char) + (features)	77.59
B-LSTM-CRF + (random)	75.05
B-LSTM-CRF + (random) + (features)	77.81
B-LSTM-CRF + (cc)	82.85
B-LSTM-CRF + (cc) + (char)	83.35
B-LSTM-CRF + (cc/mimic) + (char)	82.70
B-LSTM-CRF + (cc/mimic) + (char) + (features)	83.29

Model	DrugBank F1 score (%)	MedLine F1 score (%)
(b) DNR results over the DrugBank and MedLine datasets		
WBI-NER [3]	87.80	58.10
Hybrid-DDI [2]	80.00	37.00
Word2Vec + DINTO [1]	75.00	57.00
CRF + (random)	28.70	13.65
CRF + (features)	44.52	20.19
CRF + (cc)	43.42	32.62
CRF + (cc/mimic)	53.12	30.87
CRF + (cc/mimic) + (features)	66.45	29.36
B-LSTM + (random)	65.09	21.28
B-LSTM + (random) + (features)	75.43	30.88
B-LSTM + (cc)	71.75	42.39
B-LSTM + (cc) + (char)	84.35	43.33
B-LSTM + (cc/mimic) + (char)	83.63	44.39
B-LSTM + (cc/mimic) + (char) + (features)	84.06	45.92
B-LSTM-CRF + (random)	69.50	44.60
B-LSTM-CRF + (random) + (features)	75.78	43.36
B-LSTM-CRF + (cc)	79.03	57.87
B-LSTM-CRF + (cc) + (char)	87.87	59.02
B-LSTM-CRF + (cc/mimic) + (char)	88.38	60.66
B-LSTM-CRF + (cc/mimic) + (char) + (features)	87.42	59.75

Table 4
Percentage of words initialized with pre-trained embeddings in the train, dev and test of the respective datasets.

	Common Crawl (cc)	Common Crawl + MIMIC-III (cc/mimic)
i2b2/VA	99.99%	99.99%
DrugBank	49.50%	67.02%
MedLine	49.10%	61.51%

Table 5
Results by class for the B-LSTM-CRF with character-level and cc/mimic embeddings.

	Entities	i2b2/VA		
		Precision	Recall	F1 score
(a) i2b2/VA				
B-LSTM-CRF + (cc) + (char)	problem	81.29	83.62	82.44
	test	84.74	85.01	84.87

	Entities	DrugBank			MedLine		
		Precision	Recall	F1 score	Precision	Recall	F1 score
(b) SemEval-2013 Task 9.1							
B-LSTM-CRF + (cc/mimic) + (char)	group	81.69	87.88	84.67	69.14	60.22	64.37
	drug	94.77	89.56	91.83	73.89	77.33	75.57
	drug_n	00.00	00.00	00.00	68.18	25.57	37.19

that it may not be able to achieve it at all. In particular, we see that the CRF has performed the worst with *MedLine*. A possible explanation can be found in the "curse of dimensionality": *MedLine* is a small dataset (1627 training sentences), while the overall dimensionality of the input embeddings is 746. This makes the learning problem very sparse and seems to seriously affect a linear model such as the CRF. On the contrary, the non-linear internal architecture of the neural networks may in some cases help reduce the effective dimensionality and mollify this problem.

5.4.3. Accuracy by entity classes

Table 5a and b break down the results by entity class for the best model on each dataset. With the *MedLine* dataset, we can notice the poor performance at detecting *brand*. In *DrugBank*, the same issue occurs with entity class *drug_n*. This issue is likely attributable to the small sample size. Instead, the *i2b2/VA* dataset all entity classes are detected with similar F1 scores, likely owing to the larger number of samples per class. However, we see that brand achieves the second best F1-score in *DrugBank* despite its relatively low frequency in the dataset, and that *drug_n* obtains a very poor performance in *MedLine* even if it has the second highest frequency. We identify two other main factors that may have a major impact on the accuracy: (1) the average length of the entities in each class, and (2) the number of test entities that had not been seen during the training stage. In this respect, the *brand* and *drug* entities are usually very short (average ~ 1 word), while the *group* and *drug_n* entities often have multiple words. Since shorter entities are easier to predict correctly, *brand* obtains better accuracy than *group* in *DrugBank*. On the other hand, the *drug_n* and *group* entities have similar length, but in *MedLine drug_n* obtains a very poor performance. This is most likely because no entity of type *drug_n* that appears in the test set had been seen during training. Conversely, a large percentage of the test *group* entities had been seen during training and have therefore proved easier to predict.

6. Conclusion

In this paper, we have set to investigate the effectiveness of the Bidirectional LSTM and Bidirectional LSTM-CRF – two specific architectures of recurrent neural networks – for drug name recognition and clinical concept extraction, and compared them with a baseline CRF model. As input features, we have applied combinations of different word embeddings (Common Crawl and MIMIC–III), character-level embedding and conventional feature engineering. We have showed that the neural network models have obtained significantly better results than the CRF, and reported state-of-the-art results over the *i2b2/VA*, *DrugBank* and *MedLine* datasets using the B-LSTM-CRF model. We have also provided evidence that retraining GloVe on a domain-specific dataset such as MIMIC-III can help learn vector representations for domain-specific words and increase the classification accuracy. Finally, we have showed that adding hand-crafted features does not further improve performance since the neural networks can learn useful word representations automatically from pre-trained word embeddings. Consequently, time-consuming, domain-specific feature engineering can be usefully avoided.

Conflict of interest

The authors of this work do not have any kind of conflict of interests.

References

[1] I. Segura-Bedmar, V. Suarez-Paniagua, P. Martinez, Exploring word embedding for drug name recognition, in: 6th International Workshop on Health Text Mining and Information Analysis (LOUHI), 2015, p. 64.
[2] A.B. Abacha, M.F.M. Chowdhury, A. Karanasiou, Y. Mrabet, A. Lavelli, P. Zweigenbaum, Text mining for pharmacovigilance: using machine learning for drug name recognition and drug-drug interaction extraction and classification, J. Biomed. Inform. 58 (2015) 122–132.
[3] T. Rocktaschel, T. Huber, M. Weidlich, U. Leser, WBI-NER: the impact of domain specific features on the performance of identifying and classifying mentions of drugs, in: 7th International Workshop on Semantic Evaluation, 2013, pp. 356–363.
[4] B. deBruijn, C. Cherry, S. Kiritchenko, J. Martin, X. Zhu, Machine-learned solutions for three stages of clinical information extraction: the state of the art at i2b2 2010, J. Am. Med. Inform. Assoc. 18 (5) (2011) 557–562.
[5] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, C. Dyer, Neural architectures for named entity recognition, North American Chapter of the Association for Computational Linguistics (NAACL) (2016).
[6] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A.R. Mohamed, N. Jaitly, et al., Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, IEEE Signal Process. Mag. 29 (6) (2012) 82–97.
[7] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inform. Process. Syst. (NIPS) (2012) 1097–1105.
[8] J. Pennington, R. Socher, C.D. Manning, Glove: global vectors for word representation, Empir. Methods Nat. Lang. Process. (EMNLP) 14 (2014) 1532–1543.
[9] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, Adv. Neural Inform. Process. Syst. (NIPS) (2013) 3111–3119.
[10] A.E. Johnson, T.J. Pollard, L. Shen, L.W. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L.A. Celi, R.G. Mark, MIMIC-III, a freely accessible critical care database, Sci. Data 3 (2016).
[11] R. Chalapathy, E.Z. Borzeshi, M. Piccardi, An investigation of recurrent neural architectures for drug name recognition, in: 7th International Workshop on Health Text Mining and Information Analysis (LOUHI), 2016.
[12] R. Chalapathy, E.Z. Borzeshi, M. Piccardi, Bidirectional LSTM-CRF for clinical concept extraction, Clinical Natural Language Processing Workshop (ClinicalNLP) (2016).
[13] S. Liu, B. Tang, Q. Chen, X. Wang, X. Fan, Feature engineering for drug name recognition in biomedical texts: feature conjunction and feature selection, Comput. Math. Methods Med. (2015) 1–9.
[14] W. Boag, K. Wacome, T. Naumann, A. Rumshisky, Cliner: a lightweight tool for clinical named entity recognition, AMIA Joint Summits on Clinical Research Informatics (poster) (2015).
[15] R. Lebret, R. Collobert, Word emdeddings through hellinger PCA, European Chapter of the Association for Computational Linguistics (EACL) (2013).
[16] A. Nikfarjam, A. Sarker, K. O'Connor, R. Ginn, G. Gonzalez, Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features, J. Am. Med. Inform. Assoc. (2015).
[17] Y. Wu, J. Xu, M. Jiang, Y. Zhang, H. Xu, A study of neural word embeddings for named entity recognition in clinical text, AMIA Ann. Symp. Proc. (2015).
[18] F. Dernoncourt, J.Y. Lee, O. Uzuner, P. Szolovits, De-identification of patient notes with recurrent neural networks, J. Am. Med. Inform. Assoc. 156 (2016).
[19] A. Cocos, A.G. Fiks, A.J. Masino, Deep learning for pharmacovigilance: recurrent neural network architectures for labeling adverse drug reactions in Twitter posts, J. Am. Med. Inform. Assoc. 180 (2017).
[20] J. Xie, X. Liu, Zeng D. Dajun, Mining e-cigarette adverse events in social media using Bi-LSTM recurrent neural network with word embedding representation, J. Am. Med. Inform. Assoc. 45 (2017).
[21] Q. Wei, T. Chen, R. Xu, Y. He, L. Gui, Disease named entity recognition by combining conditional random fields and bidirectional recurrent neural networks, Database (2016).
[22] A.N. Jagannatha, H. Yu, Structured prediction models for RNN based sequence labeling in clinical text, Conference on Empirical Methods in Natural Language Processing (EMNLP) 2016 (2016) 856.

[23] M. Gridach, Character-level neural network for biomedical named entity recognition, J. Biomed. Inform. 70 (2017) 85–91.
[24] J.Y. Lee, F. Dernoncourt, O. Uzuner, P. Szolovits, Feature-augmented neural networks for patient note de-identification, Clinical Natural Language Processing Workshop (ClinicalNLP) (2016).
[25] J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, International Conference on Machine Learning (ICML) 1 (2001) 282–289.
[26] HCRF. Available from: <http://multicomp.ict.usc.edu/?p=790>.
[27] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Trans. Neural Networks 5 (2) (1994) 157–166.
[28] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[29] GloVe. Available from: <https://nlp.stanford.edu/projects/glove/>.
[30] Ö. Uzuner, B.R. South, S. Shen, S.L. DuVall, 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text, J. Am. Med. Inform. Assoc. 18 (5) (2011) 552–556.
[31] M. Herrero-Zazo, I. Segura-Bedmar, P. Martínez, T. Declerck, The DDI corpus: An annotated corpus with pharmacological substances and drug–drug interactions, J. Biomed. Inform. 46 (5) (2013) 914–920.
[32] D. Nadeau, S. Sekine, A survey of named entity recognition and classification, Lingvisticae Investigationes 30 (1) (2007) 3–26.
[33] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, Y. Bengio, Theano: a CPU and GPU math compiler in Python, in: Proc. 9th Python in Science Conf. 2010, pp. 1–7.
[34] HealthNER. Available from: <https://github.com/ijauregiCMCRC/healthNER>.
[35] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (2012) 281–305.
[36] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[37] X. Fu, S. Ananiadou, Improving the extraction of clinical concepts from clinical records, in: Proceedings of BioTxtM14 Workshop, 2014.