Feature Importance for Biomedical Named Entity Recognition

Hamish Huggard[1], Aaron Zhang[1], Edmond Zhang[2], Yun Sing Koh[1]

[1] University of Auckland, Auckland 1010, New Zealand
{hhug934, azha065}@aucklanduni.ac.nz
ykoh@cs.auckland.ac.nz
[2] Orion Health, 181 Grafton Rd, Grafton, Auckland 1010, New Zealand
edmond.zhang@orionhealth.com

Abstract. Within the domain of biomedical natural language processing (bioNLP), researchers have used many token features for machine learning models. With recent progress in word embeddings algorithms, it is no longer clear if most of these features are still useful. In this paper we survey the features which have been used in bioNLP, and evaluate each feature's utility in a sample bioNLP task: the N2C2 2018 named entity recognition challenge. The features we test include two types of word embeddings, syntactic, lexical, and orthographic features, character-embeddings, and clustering and distributional word representations. We find that using fastText word embeddings results in a significantly higher F_1 score than using any other individual feature (0.9142 compared to 0.8750 for the next-best feature). Furthermore, we conducted several experiments using combinations of features, and found that all tested combinations attained a lower F_1 score than using word embeddings only. This indicates that supplementing word embeddings with additional features is not beneficial, and may even be detrimental.

Keywords: natural language processing · biomedical NLP · word embeddings · feature importance · named entity recognition

1 Introduction

To achieve biomedical natural language processing (biomedical NLP, or bioNLP) tasks, such as summarising clinical narratives, question answering about patient history, and extracting relations between entities in a health records, many features are conceivably useful[3] For a given token, an algorithm might benefit from any of the following: the syntactic properties of the token, whether the token names a known drug or disease, the orthographic properties of the token, which other tokens tend to co-occur with the token, and so on. At various times, bioNLP researchers have used many features which attempted to capture this kind of information. However with recent advances in word embedding algorithms, it is now conceivable that all previously employed features are now obsolete. Word

[3] Our code is available at https://github.com/lajesticvantrashell/N2C2_2018_track_2.

embeddings are mappings from tokens to fixed dimensional vectors. The mappings should ideally reflect semantic or syntactic information about tokens, so that the vector corresponding to “king” should be similar to that of “monarch”. Current word embeddings can take into account semantic, syntactic, morphological [1], and even contextual information [15] about tokens, and have been used to achieve state of the art in several tasks [15]. If it could in fact be verified that current word embedding algorithms render other token features moot, this would be very valuable knowledge for bioNLP researchers. It would simplify the feature engineering task to merely finding the best word embedding.

As a first step towards investigating whether current word embedding techniques obviates the use of other features, we investigate the utility of common bioNLP features in Track 2 of the 2018 National NLP Clinical Challenges (N2C2). This is a named entity recognition (NER) task, in which entities in free medical text relating to drugs and adverse drug events (ADEs) must be annotated. An adverse drug event is defined as an “adverse outcome that occurs while a patient is taking a drug” [5]. Because NER is one of the simplest NLP tasks, this challenge is a natural first domain to investigate feature importance. Our contributions are two-fold. First, we provide a survey of the main features used in bioNLP, with implementations of each feature in our accompanying code. Secondly, we have conducted a feature importance study in which we trained and tested a simple neural network model using each of these features individually on the N2C2 2018 NER challenge. We found that word embeddings produced a significantly higher precision, recall, and F_1 score than any other feature (F_1 of 0.9142 compared to 0.8750 for the next-best feature). We also evaluated several sets of features, and found that supplementing word embeddings with additional features is not beneficial.

This paper is organised as follows. Section 2 discusses related work. Section 3 gives details of our feature importance study method. Section 4 describes the features we use in detail. Section 5 describes the results of our experiments. Section 6 concludes this investigation.

2 Related Work

Many feature engineering or feature importance experiments have been done in the bioNLP domain which used word embeddings. Tang et al. compared several types of word representation features on bioNLP tasks, including Brown clusters, word embeddings, and random indexing [18]. None of these representations was clearly superior to the others, but the combination of all three produced the best results. Liu et al. found that a word embedding feature could achieve better results at drug NER than a traditional drug lexicon-based system [11]. However, the combination of lexical and word embedding features did better than both and achieved state of the art results. Recently, Chen et al. developed an ADE NER system which benefited from lexical features [4]. Much of this research suggests that although word embeddings are useful features, the combination of word embeddings with other conventional features tends to produce the best results.

However, word embedding algorithms have improved significantly since these previous studies were conducted, and it is worth re-examining their conclusions.

Some work has also been done on the best approach to employing word embeddings for bioNLP tasks. Muneeb et al. evaluated different word embedding algorithms on semantic similarity and relatedness tasks in a biomedical context [13], finding that the skipgram implementation of the word2vec was the most accurate. Wang et al. [20] and Wu et al. [23] both compared the performance of word embeddings trained on biomedical corpora versus general corpora on downstream bioNLP tasks. In general, word embeddings trained on biomedical corpora outperformed those trained on general corpora.

It is worth noting that in most actual biomedical NLP applications rule-based systems still dominate [21]. However, given the significant role of neural networks and word embeddings in state-of-the-art natural language processing [12][15], this seems likely to change in the future. We therefore focus on neural network models.

3 Feature Importance Experiment Methodology

Our feature importance study began by dividing the official N2C2 2018 Track 2 training dataset into a training/validation/development partition, which was fixed throughout the study. Then, for each feature we describe in Section 4, we trained a simple neural network model on the training set using only that feature, and report the lenient precision, recall, and micro F_1 score of the model on the development set. We also performed a single evaluation of our best feature set on the official N2C2 test set, so that our results could be compared to the official competition outcomes.

3.1 Model Architecture

Because we were primarily interested in studying feature importance, and not creating the most accurate model possible, we elected to use a very simple neural network model for our experiments. However, preliminary experiments with more complex neural network architectures showed no discernible difference in F_1 score. The text was processed in windows of tokens. The features for each token were generated and then concatenated into a single vector per token. These vectors were then fed into a bidirectional LSTM (bi-LSTM). A dense layer with softmax activation then predicted token labels from the bi-LSTM outputs. This simple architecture is illustrated in Figure 1 and was implemented in `Keras` with `Tensorflow` backend.

A dropout rate of 0.3 was used between layers. In each experiment the model was trained using the Nesterov ADAM optimiser with a batch size of 128. The models were trained for 40 epochs, using early stopping with a patience of 5 epochs. The classification error was given by categorical cross entropy.

We did not perform a formal hyperparameter optimisation for this architecture. However, preliminary testing indicated that the following were sensible options: a token window size of 128, and a latent dimension for the bi-LSTM of 128. At training time the token window had a slide of 32, at test time it was equal to the window size.

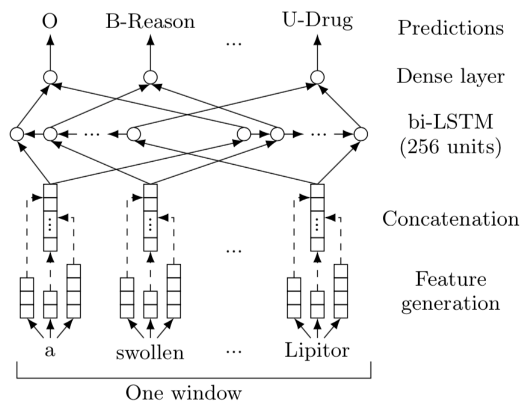

Fig. 1. Our architecture.

Table 1. Effect of training corpora on fastText.

Corpus	Prec.	Recall	F_1
Wikipedia	0.9200	0.8472	0.8821
Pubmed	0.9085	0.8681	0.8878
Wiki-news	**0.9354**	0.8467	0.8889
PMC	0.9312	0.8742	0.9018
MIMIC	0.9353	0.8764	0.9049
Combined	0.9353	**0.8941**	**0.9142**

Table 2. Effect of cluster number on Brown clustering.

#Clusters	Prec.	Recall	F_1
100	0.8933	0.7119	0.7923
200	0.8883	0.7772	0.8290
500	0.9079	0.7824	0.8405
1000	**0.9086**	**0.8231**	**0.8638**

3.2 Dataset

Our models were evaluated using a dataset of nearly 505 clinical narratives provided by Track 2 of the 2018 National NLP Clinical Challenges (N2C2)[4] . The N2C2 dataset is a drawn from the MIMIC-III (Medical Information Mart for Intensive Care III) database [7], with annotations added by domain experts. The annotations consist of entity tags indicating the presence of drug and ADE information. Specifically, the named entities for the N2C2 challenge are “Drug”, “Reason”, “Strength”, “Frequency”, “ADE”, “Dosage”, “Duration”, “Form” and “Route”.

We used BILOU segment representation for multi-token entities. Under BILOU, for each named entity which spans multiple tokens, the beginning token is labelled with a “B”, the last token with an “L”, and those tokens inside the named entity (i.e., between the beginning and last token) with an “I”. Tokens which are outside of all named entities are labelled with an “O”, and named entities consisting of a single (unique) token are labelled “U”. So for example, we get the following segment representation: “treated/O with/O vanc/U-Drug for/O the/O septic/B-Reason left/I-Reason knee/L-Reason”.

[4] `https://portal.dbmi.hms.harvard.edu/projects/n2c2-t2/`

For training certain features we also made use of the following biomedical corpora. The MIMIC III dataset [7] contains de-identified information on 40,000 patient encounters in a critical care unit. We extracted the the patient summaries and used these to pre-train some of our features. The PubMed Central (PMC) dataset is a repository of biomedical and life sciences journal literature. We downloaded a portion of the articles from the repository for Dec 2018 which contained around 200K articles. The Pubmed dataset is a free search engine primarily accessing the MEDLINE bibliographic database of life sciences and biomedical information. We extracted over 200K articles and abstracts in a snapshot we took by searching for relevant clinical articles or abstracts.

3.3 Evaluation

We evaluated the performance of our models using the official N2C2 track 2 evaluation script. The primary evaluation metric of the competition is lenient micro F_1 (henceforth simply "F_1"), although we also report the precision and recall. "Lenient" here meaning that any overlap in the predicted and true span of a named entity is counted as a correct prediction.

The data is split into 303 official training documents and 202 official test documents. Because we tested several models, to avoid overfitting to the test data and positively biasing our F_1 score, we further subdivided the training data into training, validation, and development dataset, with an 8/1/1 split. The validation set is used for early stopping, and the development set is used for F_1 score evaluation. As a final step in our experiments, we evaluated our most accurate model (according to the development dataset) on the official test set.

4 Features

In this section we list all the features we experimented with in our experiments, aggregated from the literatures on biomedical NLP [18][11][10][3], and general NER [14]. First we considered word representation (WR) features, which can be categorised as word embeddings, cluster-based representations, and distributional representations. We employed two word embedding algorithms: fastText [1] and ELMo [15]. We used Brown clustering as a cluster-based WR, and random indexes as a distribution-based WR. In addition to WRs, we also experimented with character-embedding features, orthographic features, lexical features, and syntactic features.

4.1 FastText

FastText is a word embedding algorithm based on the skipgram model, primarily distinguished from preceding word embedding algorithms in two ways: first, it's computationally much more efficient, and secondly, it generates embeddings at the character n-gram model, as well as at the token level. This allows it to model word morphology, and determine embeddings for tokens which did not appear in the training data. For our experiments, we downloaded two pre-trained fastText embeddings provided by the original authors. Both are trained on general corpora: the first on Wikipedia[5], and the second on both Wikipedia and Google News[6]. We also trained new embeddings on the biomedical corpora MIMIC-III, Pubmed, PMC, and the combination of the three. The results of using fastText embeddings trained on each of these corpora are given in Table 1.

4.2 ELMo

Embeddings from Language Models (ELMo) is a word embedding algorithm which recently debuted by significantly improving the state of the art in six difficult NLP tasks [15]. ELMo is a deep bidirectional language model (biLM), meaning it is composed of several bi-LSTMs stacked on top of one another (imagine Figure 1 but with the bi-LSTM layer repeated several times). The final word embedding for a token is a learned linear combination of all the LSTM states in a vertical stack. This approach allows ELMo to resolve polysemy - words with multiple meanings - by taking context into account. Due to resource constraints, we did not fine tune ELMo to our task, and simply used out-of-the-box ELMo with TensorFlow Hub[7] to generate 1024-dimensional word embeddings for each token.

4.3 Brown Clusters

Brown hierarchical word clustering is an algorithm designed to allocate classes to tokens, such that the average mutual information between adjacent classes in text is maximised [2]. Because the clustering is hierarchical, clusters may be represented by binary strings denoting the binary tree traversal from the root to the leaf representing the cluster. This is illustrated in Figure 2, in which we can see that “with” is more closely related to “between” than to “in”.

We ran an implementation of Brown clustering [9] over the pre-tokenised MIMIC-III corpus. The number of clusters is a hyper-parameter to be tuned. We considered four different numbers of clusters, and the results of each cluster number are given in Table 2. 1000 clusters were best in our experiments. Examples of the resulting clusters are shown in Table 4. The cluster feature is zero-padded to a fixed length.

4.4 Random Indexing

Random indexing is a distributional word representation method, so tokens which tend to co-occur in documents will have similar representations. Many approaches to distributional word representation - such as latent semantic analysis - rely on a costly dimension reduction step, which must be repeated every

[5] `https://fasttext.cc/docs/en/pretrained-vectors.html`
[6] `https://fasttext.cc/docs/en/english-vectors.html`
[7] `https://tfhub.dev/google/elmo/1`

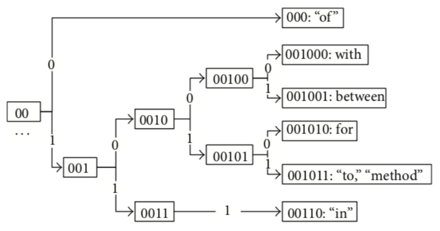

Fig. 2. Brown clusters illustration [18].

Table 3. Example of syntactic tags produced by the Genia Tagger.

tokens	POS	chunk
The	Determiner.	Noun phrase
patient	Noun, singular.	Noun phrase
was	Verb, past tense.	Verb phrase
tested	Verb, past part.	Verb phrase

Table 4. Illustration of several features with the text "2.50 mg - Lasix".

tokens	jochem	clusters	shape	short_shape	len	title	upper	lower	numeric	symbol
2.50	False	11110101	0!00	0!0	4	False	0.00	0.00	0.75	0.25
mg	True	1111011	xx	x	2	False	0.00	1.00	0.00	0.00
-	False	11110111101	!	!	1	False	0.00	0.00	0.00	1.00
Lasix	True	1111010000111	Xxxxx	Xx	5	True	0.20	0.80	0.00	0.00

time the representation is updated with a new document. Random indexing gets around this problem using the fact that random pairs of sparse, high-dimensional vectors are on average almost orthogonal. This allows a dimension-reduction computation to be approximated by accumulating such vectors onto token representations.

In particular, we adopted the random indexing method described in [16]. Each document D_j was initially assigned an n-dimensional index vector r_j. Index vectors are high dimensional and most elements are zero, with a few elements being +1 or -1. Each element of the vector was given by

$$r_{j,i} = \sqrt{s} \begin{cases} -1 & \text{with probability } \frac{1}{2s} \\ 0 & \text{with probability } 1 - \frac{1}{s} \\ 1 & \text{with probability } \frac{1}{2s} \end{cases} \tag{1}$$

The next step was to generate an n-dimensional context vector v_k for each token w_k in the vocabulary. Initially, $v_k = \mathbf{0}$, and then for each occurrence of w_k in D_j, r_j was added to v_k. In this case we chose $n = 100$, and $s = \sqrt{n} = 10$. These values are indicated to be appropriate in [16]. We trained our random index representation on MIMIC-III.

4.5 Character Embeddings

To capture morphological information about tokens, we used a character-level token representation. Specifically, we used a neural network architecture called CharWNN, which was introduced in [17].CharWNN produces a representation of each token by convolving over a sequence of character embeddings, and then taking the maximum value at each index of the resulting vectors. We omit the mathematical details here, but we use the same architecture and hyperparameters as [17].

4.6 Lexical Features

Following [11], we checked each token against three popular drug lexica: drugs@FDA[8], DrugBank (version 5.1.2, released 2018-12-20) [22], and Jochem (Erasmus ontollogy) [6]. Each lexicon has a corresponding binary feature, where a value of 1 indicates that the token is present in the lexicon, and 0 indicates it is not. Examples of this token can be seen in Table 4. Each of the entries in these lexica were preprocessed as follows. The entries were tokenised using `Spacy`, and then each token was lower-cased and stemmed. Any tokens with no alphabetic characters were removed. With this process, 6799 tokens were extracted from drugs@FDA, 14317 tokens were extracted from the 'drug name' and 'active ingredients' columns of the Drugbank database, and 941334 tokens were extracted from Jochem. In addition to a binary feature for each lexicon, we also considered the concatenation of these features as a 3-dimensional feature which we called "all lexica".

4.7 Orthographic Features

We used eight orthographic features to encode information about the *kinds* of characters present in the word. These features are illustrated in Table 4. Firstly, we used a simple orthography feature which gave the proportion of characters in the token which were numeric, uppercase alphabetic, lowercase alphabetic, and symbols, plus a Boolean of whether the token was in title case. We can get a more fine grained picture of the token's orthography by also considering the ordering of character types. This was achieved with word shapes. To obtain the word shape of a token, all uppercase letters were substituted with "X", all lowercase letters were substituted with "x", all numerals were substituted with "0", all symbols were substituted with "!", and all whitespace characters were substituted with "". In addition to the word shapes, short shapes are another feature which summarise the order in which types of characters appear in the token. These were produced by removing consecutive duplicates of the same character type from the word shape. Because there are an unlimited number of possible word shapes and short shapes, all but the 99 most common shapes were replaced with the empty shape "", so that the shape can be encoded in a 100-dimensional one-hot vector.

4.8 Syntactic Features

For syntactic features we used the Genia toolkit[9] to generate part of speech (POS) and chunk tags for each token. Examples of these features are given in Table 3. The Genia toolkit is trained on biomedical texts and has reported high accuracy on several bioNLP tasks [19]. The POS tags used by the Genia toolkit

[8] `https://www.fda.gov/drugs/drug-approvals-and-databases/drugsfda-data-files`

[9] `http://www.nactem.ac.uk/GENIA/tagger/`

are a variant of the Penn Treebank tags, including 38 POS tags, and 9 other tags for punctuation and symbols [8]. The chunking tags indicate where in a syntactic chunk a token resides. These tags combine BIO segment representation (which is the same as BILOU representation, but with B in place of U, and I in place of L), with 28 chunk-type labels. Because the range of possible tags for POS and chunks is relatively small, the tags were simply one-hot encoded.

4.9 Features Not Investigated

In our literature review we encountered several features which we did not investigate - either because they did not seem important enough, or because we could not figure out how to implement them with our architecture. We did not explicitly include any character n-grams, such as suffixes or prefixes. However, the character embedding should in principle capture the same information as character n-grams. We did not use a lexicon of cues denoting contexts in which entities are likely to occur, such as "medication:" indicating that the following token is likely to be a drug. Finally, we did not perform any fuzzy matching with lexica, and we did not use phrase length.

5 Feature Importance Study Results

For each of the individual features described above, we evaluated model accuracy using *only* that feature (using the predetermined train/val/dev datasets described in Section 3). The results are given in Table 5. The two word embedding features - fastText and ELMo - achieved the two highest F_1 scores. The best word embedding feature (fastText trained on combined corpus, 0.9142) is significantly better than the highest non-word embedding feature (character embeddings, 0.8750), and even the worst character embedding feature (fastText trained on Wikipedia, 0.8821) is still better than any non-word embedding feature. These and similar observations suggest that the different types of features can be ranked as followed: word embbeddings are the strongest features, followed by other word representations, orthographic features and POS tags come next, and lexical features and chunk tags are the weakest features (although lexical features also have the smallest domains). Drugbank in particular produced a remarkably low F_1 score.

Due to resource constraints we were not able to run a feature selection algorithm over the full set of features. However, we did evaluate the model when the full set of features are used, when only the top-5 most successful features were used, and when the top-2 features were used. We did the latter in two different ways: first, we considered fastText and ELMo as different features, and so used both of them in the top-2 experiment. Second, we considered fastText and ELMo as variations of a "word embedding" feature, and so used only fastText, plus the best non-word embedding feature: character embeddings. Every combination of features resulted in a lower precision, recall, and F_1 score than fastText (combined corpus) by itself. This indicates that not only are word embeddings the most useful feature for this task, but also that supplementing word embeddings with other features is not useful, and is perhaps even detrimental.

To ground our results in the official N2C2 competition outcomes, we also tested our best feature set on the official competition test data. For this final experiment, we kept the previous window size of 128, but reduced the window slide to 16. This is effectively a data augmentation technique which increases the size of the training data, thereby making the final model more accurate, but doubling the memory requirements and training time. For this experiment, the development dataset was added to the training dataset, but otherwise all experimental details were kept the same. The best feature set on the validation set was fastText (combined corpus) only. The final F_1 score achieved with this method was 0.9240. For comparison, the state of the art in this task is 0.9418.

Table 5. Performance of models using subsets of the features. The domains of each of each feature are also included. Binary features are denoted $\mathbb{Z}_2$, n-dimensional real-valued vector features are denoted $\mathbb{R}^n$, one-hot encoded features with n possible values are denoted $\mathbb{Z}_n$.

Feature	Domain	Precision	Recall	F_1
FastText (Combined)	$\mathbb{R}^{100}$	**0.9353**	**0.8941**	**0.9142**
ELMo	$\mathbb{R}^{1024}$	0.9190	0.8610	0.8890
Character Embedding	$\mathbb{R}^{50}$	0.9162	0.8374	0.8750
Brown Clusters	$\mathbb{Z}_2^{15}$	0.9086	0.8231	0.8638
Random Index	$\mathbb{R}^{100}$	0.8551	0.7465	0.7971
Orthography	$\mathbb{R}^5 \times \mathbb{Z}_2$	0.9069	0.6006	0.7226
POS	$\mathbb{Z}_{47}$	0.8699	0.5878	0.7016
Word shape	$\mathbb{Z}_{100}$	0.8959	0.5664	0.6941
Short shape	$\mathbb{Z}_{100}$	0.8588	0.5438	0.6660
Length	$\mathbb{Z}$	0.8524	0.5249	0.6497
All lexica	$\mathbb{Z}_2^3$	0.8286	0.5166	0.6364
Drugs@FDA	$\mathbb{Z}_2$	0.7609	0.4173	0.5390
Chunk	$\mathbb{Z}_{27}$	0.8348	0.3798	0.5220
Jochem	$\mathbb{Z}_2$	0.8503	0.3572	0.5030
Drugbank	$\mathbb{Z}_2$	0.5672	0.0425	0.0791
FastText+CE	$\mathbb{R}^{350}$	0.9330	0.8855	0.9086
FastText+ELMo	$\mathbb{R}^{1324}$	0.9329	0.8777	0.9044
Top-5	$\mathbb{R}^{1489}$	0.9192	0.8833	0.9009
All features	$\mathbb{R}^{1873}$	0.9186	0.8848	0.9014
FastText (test set)	$\mathbb{R}^{300}$	**0.9476**	**0.9016**	**0.9240**

6 Conclusion

In accord with past research [20][23], our experiments indicated that word embeddings trained on biomedical corpera are more useful for bioNLP tasks than those trained on general corpera; as is visible in Table 1. Unlike past research, which found that conjunctions of word embeddings with other word representations were the best feature sets for BioNLP tasks [18][11][4], our experiments indicated that best results are obtained by using word embeddings *only*. This discrepancy can be accounted for by the fact that significant progress has been made in word embedding algorithms since these early studies were conducted. These results lend support to the hypothesis that by using state-of-the-art word embedding algorithms, other features which have traditionally been used for bioNLP - such as explicitly syntactic, morphological, and contextual features - can be made redundant. This greatly simplifies the task of bioNLP practitioners, whose feature engineering task is then simplified to merely finding a good word embedding.

There are several caveats to this conclusion. Our experiments only investigated the effect of using different features for a simple neural network model with a single architecture and hyperparameters. Furthermore, we only evaluated the model on a single NER bioNLP task. Further research is therefore required to validate our findings in the broader bioNLP context. On the other hand, our experimental procedure is quite representative of a typical bioNLP scenario: NER is a standard NLP benchmark, and simple recurrent networks - especially biLSTMs - are a popular model choice for natural language problems. Furthermore, in our preliminary experiments we also tested a more complex neural network model, and found very similar results to those reported here.

Acknowledgements

This research was supported by the Precision Driven Health Partnership (www.precisiondrivenhealth.com).

References

1. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
2. Brown, P.F., Desouza, P.V., Mercer, R.L., Pietra, V.J.D., Lai, J.C.: Class-based n-gram models of natural language. Computational linguistics **18**(4), 467–479 (1992)
3. Campos, D., Matos, S., Oliveira, J.L.: Biomedical named entity recognition: a survey of machine-learning tools. In: Theory and Applications for Advanced Text Mining. IntechOpen (2012)
4. Chen, Y., Zhou, C., Li, T., Wu, H., Zhao, X., Ye, K., Liao, J.: Named entity recognition from chinese adverse drug event reports with lexical feature based bilstm-crf and tri-training. Journal of biomedical informatics **96**, 103252 (2019)
5. Edwards, I.R., Aronson, J.K.: Adverse drug reactions: definitions, diagnosis, and management. The lancet **356**(9237), 1255–1259 (2000)
6. Hettne, K.M., Stierum, R.H., Schuemie, M.J., Hendriksen, P.J., Schijvenaars, B.J., Mulligen, E.M.v., Kleinjans, J., Kors, J.A.: A dictionary to identify small molecules and drugs in free text. Bioinformatics **25**(22), 2983–2991 (2009)
7. Johnson, A.E., Pollard, T.J., Shen, L., Li-wei, H.L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L.A., Mark, R.G.: Mimic-iii, a freely accessible critical care database. Scientific data **3**, 160035 (2016)

8. Kim, J.D., Ohta, T., Teteisi, Y., Tsujii, J.: Genia corpus manual. Tech. rep., Citeseer (2006)
9. Liang, P.: Semi-supervised learning for natural language. Ph.D. thesis, Massachusetts Institute of Technology (2005)
10. Liu, F., Chen, J., Jagannatha, A., Yu, H.: Learning for biomedical information extraction: Methodological review of recent advances. arXiv preprint arXiv:1606.07993 (2016)
11. Liu, S., Tang, B., Chen, Q., Wang, X.: Effects of semantic features on machine learning-based drug name recognition systems: word embeddings vs. manually constructed dictionaries. Information **6**(4), 848–865 (2015)
12. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
13. Muneeb, T., Sahu, S., Anand, A.: Evaluating distributed word representations for capturing semantics of biomedical concepts. Proceedings of BioNLP 15 pp. 158–163 (2015)
14. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**(1), 3–26 (2007)
15. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)
16. QasemiZadeh, B., Handschuh, S.: Random indexing explained with high probability. In: International Conference on Text, Speech, and Dialogue. pp. 414–423. Springer (2015)
17. dos Santos, C.N., Zadrozny, B.: Learning character-level representations for part-of-speech tagging. In: Proceedings of the 31st International Conference on Machine Learning (ICML-14). pp. 1818–1826 (2014)
18. Tang, B., Cao, H., Wang, X., Chen, Q., Xu, H.: Evaluating word representation features in biomedical named entity recognition tasks. BioMed research international **2014** (2014)
19. Tsuruoka, Y., Tateishi, Y., Kim, J.D., Ohta, T., McNaught, J., Ananiadou, S., Tsujii, J.: Developing a robust part-of-speech tagger for biomedical text. In: Panhellenic Conference on Informatics. pp. 382–392. Springer (2005)
20. Wang, Y., Liu, S., Afzal, N., Rastegar-Mojarad, M., Wang, L., Shen, F., Kingsbury, P., Liu, H.: A comparison of word embeddings for the biomedical natural language processing. Journal of biomedical informatics **87**, 12–20 (2018)
21. Wang, Y., Wang, L., Rastegar-Mojarad, M., Moon, S., Shen, F., Afzal, N., Liu, S., Zeng, Y., Mehrabi, S., Sohn, S., et al.: Clinical information extraction applications: a literature review. Journal of biomedical informatics **77**, 34–49 (2018)
22. Wishart, D.S., Feunang, Y.D., Guo, A.C., Lo, E.J., Marcu, A., Grant, J.R., Sajed, T., Johnson, D., Li, C., Sayeeda, Z., et al.: Drugbank 5.0: a major update to the drugbank database for 2018. Nucleic acids research **46**(D1), D1074–D1082 (2017)
23. Wu, Y., Xu, J., Jiang, M., Zhang, Y., Xu, H.: A study of neural word embeddings for named entity recognition in clinical text. In: AMIA Annual Symposium Proceedings. vol. 2015, p. 1326. American Medical Informatics Association (2015)