Journal of the American Medical Informatics Association, 0(0), 2017, 1–10
doi: 10.1093/jamia/ocw180
Research and Applications

Research and Applications

Deep learning for pharmacovigilance: recurrent neural network architectures for labeling adverse drug reactions in Twitter posts

Anne Cocos, Alexander G Fiks, and Aaron J Masino

Department of Biomedical and Health Informatics, The Children's Hospital of Philadelphia Philadelphia, PA, USA

Corresponding Author: Aaron J Masino, Department of Biomedical and Health Informatics, The Children's Hospital of Philadelphia, 3535 Market St., Room 1090, Philadelphia, PA 19104, USA. E-mail: masinoA@email.chop.edu; Phone: 267-426-9148.

Received 5 August 2016; Revised 18 November 2016; Accepted 17 December 2016

ABSTRACT

Objective: Social media is an important pharmacovigilance data source for adverse drug reaction (ADR) identification. Human review of social media data is infeasible due to data quantity, thus natural language processing techniques are necessary. Social media includes informal vocabulary and irregular grammar, which challenge natural language processing methods. Our objective is to develop a scalable, deep-learning approach that exceeds state-of-the-art ADR detection performance in social media.

Materials and Methods: We developed a recurrent neural network (RNN) model that labels words in an input sequence with ADR membership tags. The only input features are word-embedding vectors, which can be formed through task-independent pretraining or during ADR detection training.

Results: Our best-performing RNN model used pretrained word embeddings created from a large, non–domain-specific Twitter dataset. It achieved an approximate match F-measure of 0.755 for ADR identification on the dataset, compared to 0.631 for a baseline lexicon system and 0.65 for the state-of-the-art conditional random field model. Feature analysis indicated that semantic information in pretrained word embeddings boosted sensitivity and, combined with contextual awareness captured in the RNN, precision.

Discussion: Our model required no task-specific feature engineering, suggesting generalizability to additional sequence-labeling tasks. Learning curve analysis showed that our model reached optimal performance with fewer training examples than the other models.

Conclusion: ADR detection performance in social media is significantly improved by using a contextually aware model and word embeddings formed from large, unlabeled datasets. The approach reduces manual data-labeling requirements and is scalable to large social media datasets.

Key words: natural language processing, neural networks (computer), adverse drug reaction, social media, Twitter messaging

BACKGROUND AND SIGNIFICANCE

Comprehensive knowledge of adverse drug reactions (ADRs) can reduce their detrimental impact on patients and the health system.[1,2] Practically, clinical trials cannot investigate all settings in which a drug will be used,[1–3] making it impossible to fully characterize the drug's adverse effect profile before its approval. Pharmacovigilance, or post-market drug safety surveillance, identifies ADRs after a drug's release. Most current pharmacovigilance activities rely on passive spontaneous reporting system databases, such as the Federal Drug Administration's Adverse Event Reporting System (FAERS).[3,4] Such systems may be limited by delayed, biased, and underreporting of events. For example, the rate of reporting serious adverse events to FAERS is estimated to be only 1–13% of actual events.[1,5]

To address passive reporting system limitations, active pharmacovigilance methods continuously analyze frequently updated data sources. Initial research examined ADR extraction from structured electronic health record (EHR) data (eg, problems list).[6,7] These studies, however, revealed that structured EHR data contain limited ADR information, thus subsequent studies examined ADRs in EHR clinical narratives.[8–11] More recent research examines social media text as a complement to EHR data.[12–21] Twitter is particularly interesting because of its large user base, demographic variability, and publicly available data. Additionally, statistically significant correlations ($P < .001$) have been identified between certain ADRs described in Twitter data and those reported in FAERS,[17] suggesting that Twitter is a viable pharmacovigilance data source.

ADR detection in social media requires automated methods to process the high data volume. The earliest studies implemented string-matching methods to identify terms that matched predefined drug and adverse event lexicons.[12–15,17] However, lexicon matching cannot distinguish whether a drug-related event describes a reaction to or indication for a medication. Additional characteristics of social media language further limit lexicon matching as an ADR detection method. For example, social media language is informal, incorporating vocabulary and phrases from the vernacular (eg, "feeling like crap"), with frequent misspellings and irregular grammar (eg, "dis aderall got me sweatin"). It also includes abbreviations (eg, "lol") and symbols (eg, emoticons) that convey semantic information. To address these challenges, some researchers treat ADR detection as a supervised machine-learning sequence-labeling problem where the learning methods may account for the surrounding context of a given input word. This is common in natural language processing, where each token (ie, contiguous character sequence, usually corresponding to a word) is labeled with a named entity tag (eg, person). For ADR detection, tokens can be labeled as *part of* an adverse event. The most successful ADR sequence-labeling efforts employ conditional random field (CRF) models.[21–24] CRFs are limited by the scope of their input in that the model considers only the target word and its neighboring words within a fixed-width window and therefore may exclude important information mentioned more distantly.

Recurrent neural networks (RNNs) may overcome this limitation. RNNs process text as a sequence of words and contain a hidden state that "remembers" an arbitrary number of previous labeling decisions that are used to label the current token. This "memory" makes RNNs well-suited to labeling tasks where individual word labels depend on the word itself and the words and labels that precede it.[25–27] RNN model variants have achieved state-of-the-art performance in part-of-speech tagging[28] and named entity recognition.[25] RNNs have been used to elucidate patient phenotypes from longitudinal clinical data,[29] but to our knowledge have not been applied specifically to labeling drug events, particularly in social media data.

OBJECTIVE

We sought to develop an RNN model that exceeds current state-of-the-art ADR detection in social media posts. We investigated a specific architecture known as a *Bidirectional Long Short-Term Memory* (BLSTM) RNN.[27,30–32] In the standard RNN model, the network's memory is equally dependent on all previous outputs. In the LSTM variant, the model learns weights for its previous output and the current input for each sequence item.[26,30] Additionally, the BLSTM[31,32] process sequences in the forward *and* backward direction, enabling it to learn dependencies in either direction. We compared the performance of several BLSTM variants to a state-of-the-art CRF model and a baseline lexicon-based method. We specifically considered ADR labeling performance on Twitter user posts.

MATERIALS AND METHODS

This study was determined to be exempt from Institutional Review Board review by The Children's Hospital of Philadelphia.

Data collection and labeling

Our dataset combines 2 Twitter datasets. The first dataset, *Twitter ADR Dataset (v1.0)*, was published previously.[21] The dataset authors collected and annotated user posts (called tweets) using search terms for 81 drugs widely used in the US market and new drugs released between 2007 and 2010.[20,21] The published data includes unique tweet identifiers but excludes the tweet text (as prohibited by the Twitter application program interface license agreement). Thus, it was necessary to obtain text from the Twitter application program interface using the identifiers. Of the 957 identifiers in the original dataset, 641 tweets were available for download at the time of this study. The drugs represented by these tweets were not collected for a specific condition and represent a broad range of potential ADRs. This data represents 76% of our complete dataset.

We supplemented this dataset with additional tweets, denoted the *ADHD Dataset,* containing at least 1 search term corresponding to 44 brand and generic drugs used for treatment of attention deficit hyperactivity disorder (ADHD). We collected these tweets between May 1, 2015, and December 31, 2015. Tweets with URLs and reposted tweets were excluded. From these, we included 203 tweets that were identified as containing a drug-related event and annotated using the guidelines published with the *Twitter ADR Dataset (v1.0).* The labeled ADHD dataset is available at https://github.com/chop-dbhi/twitter-adr-blstm.

Our complete dataset contains 844 tweets, of which 95% contain at least 1 ADR or indication mention. We maintained the train-test split as published for the Twitter ADR Dataset and randomly divided the ADHD Dataset to create training and test sets of sizes 634 (75%) and 210 (25%) tweets. The training tweets contained 647 ADRs and 71 indications, while the test tweets contained 199 ADRs and 22 indications.

We separated tweets into individual tokens using the *ark-tweet-nlp* Python module.[33,34] We converted all tokens to lowercase and replaced "at" mentions (ie, @username) with a special token. Although the RNN model can process sequences of arbitrary length, many modeling frameworks expect fixed-length inputs for batch processing. Thus, for implementation convenience, it was necessary to standardize the number of tokens in each tweet. Therefore, we padded each tweet to match the length of the longest tweet in the training data (36 tokens). For a tweet with k tokens, if $k < 36$, we appended $(36 - k)$ "<PAD>" tokens to the beginning of the tweet, where each <PAD> token was a zero vector. If a tweet in the test data had more than 36 tokens, the model was set up to truncate it to include just the first 36 tokens. As it turned out, no test tweets contained more than 36 tokens.

It was necessary to manually label tweets in the ADHD Dataset as positive or negative for containing a drug event. For event positives, the event type and text span containing the event were labeled. Two authors independently labeled all tweets. We performed consensus reconciliation of discordant assessment until 100% agreement was achieved. The published Twitter ADR Dataset included the same labels. To facilitate supervised model training and evaluation, it was necessary to generate token labels that indicated ADR membership. A standard approach for sequence-labeling tasks is to

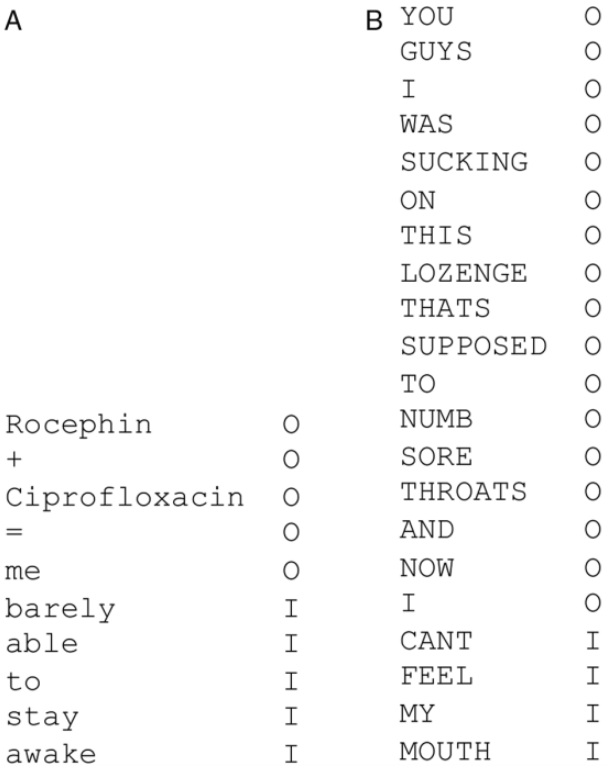

Figure 1. Examples of tweets with ADR spans labeled using our binary tag set.

use an *I-O-B* scheme, where tokens are assigned labels that indicate the token's position at the beginning (*B*), inside (*I*), or outside (*O*) an entity of interest.[35] We adopted an *I-O* scheme with 4 labels: I-ADR, I-Indication, O, and <PAD>, indicating that the given token was, respectively, part of an ADR, part of an indication, outside any ADR or indication, or was padding (Figure 1). These token labels can be automatically derived from the event type and text span labels.

BLSTM-RNN model development

Our BLSTM-RNN model processes each tweet as a sequence of tokens and predicts the label for each token. The model is called bidirectional because internally it combines 2 RNNs: a forward RNN processes the sequence from left to right, and a reverse RNN processes the sequence from right to left. The outputs of both RNNs are averaged for each token to compute the model's final label prediction.

Word embedding features

Our model takes as input an index that maps the current token to a column in a word embedding matrix. The column is a real-valued numeric representation known as a word embedding. The embeddings can be treated as fixed constants or learnable parameters. When fixed, the embedding matrix values are assumed to have been pretrained and are not updated during training. When learnable, the matrix values are treated as model parameters that are initialized with pretrained values or randomly and are updated during training. We experimented with all 3 options. For pretrained word embeddings, we adopted a published set of 400-dimensional word embeddings trained using the skip-gram algorithm on more than 400 million tweets.[28] The vocabulary of these word embeddings covered 96.5% of the tokens in our dataset. These tweets were not specific to any particular domain. Furthermore, the skip-gram algorithm[36] is an unsupervised machine-learning method designed to encode semantic and syntactic information and has no direct connection to the ADR labeling task. These word embeddings were the only input features provided to our model. Although we tested additional features, they did not produce performance differences (see Supplementary Appendix).

Model architecture

Our BLSTM-RNN model produces an ADR label for each word embedding of a length-n tweet in a process that can be described in terms of sequential steps $t = 0$ *to* $t = n - 1$ (corresponding to token position in the tweet). At step $t = i$, the forward RNN computes a D-dimensional output for token w_i and the backward RNN computes a D-dimensional output for token w_{n-1-i}. The forward and backward chains are both LSTM RNNs whose hidden layer nodes compute their output as a function of the current input (either the token embedding or the preceding hidden layer's output) and their individual outputs for the previous input. Each node contains input, forget, and output gates that weight the current input, previous hidden state, and current hidden state, respectively, when calculating its current output (see Table 1). The final hidden layer outputs of the 2 chains for a given token, w_i, are averaged element-wise and passed to a fully connected layer that reduces the D-dimensional vector to 4 values, representing probabilities for each label (<PAD>, I-ADR, I-Indication, or O) (see Figure 2).

Model variants

We evaluated 3 BLSTM variants:

- Method 1 (BLSTM-M1): Word-embedding values randomly initialized (standard normal scaled by the square root of our vocabulary size[32]) and treated as learnable parameters.
- Method 2 (BLSTM-M2): Word-embedding values initialized with a publicly available pretrained dataset[28,36] and treated as learnable model parameters.
- Method 3 (BLSTM-M3): Word-embedding values initialized as in method 2, but treated as fixed constants.

We implemented the models using the Keras Python library[37] over a Theano[38,39] backend. We used online training (1 tweet at a time) with back propagation through time,[40] for 6, 6, and 18 epochs (ie, passes over the training set) for the BLSTM-M1, BLSTM-M2, and BLSTM-M3 models, respectively. We calibrated the number of epochs for each model individually prior to training using cross-validation over the training set. Training the BLSTM-M3 model on the full 634-tweet training set for 18 epochs took ~45 min on a MacBook Pro with a 2.6 GHz processor and 16 GB RAM. Testing took under 5 s. Our model code is available at: https://github.com/chop-dbhi/twitter-adr-blstm.

Baseline sequence labeling methods

We also considered 2 baseline techniques: a lexicon-matching method and a CRF model.

Lexicon-matching method

The first baseline method identifies ADR mentions by matching concepts in a lexicon. Our lexicon, containing 13 014 concept terms, combines a previously published ADR lexicon[21] with concepts from the Consumer Health Vocabulary[41] corresponding to adverse events known to be associated with medications commonly used to treat ADHD as derived from the National Institute for Children's Health Quality Vanderbilt Assessment Follow-up—PARENT informant form. We created an Apache Lucene index from the concept terms and queried the index for concept matches within each tweet. We

Table 1. Update equations for LSTM-RNN layers

Parameters to be learned[a]	Input	Output/hidden state[b]
Input weights: $W_i, W_f, W_c, W_o \in \mathbb{R}^{D\times H}$ Recurrent weights: $U_i, U_f, U_c, U_o \in \mathbb{R}^{H\times H}$ Bias vectors: $b_i, b_f, b_c, b_o \in \mathbb{R}^{H}$	$x_t \in \mathbb{R}^{D}$	$i_t = \sigma(x_t W_i + b_i + s_{t-1} U_i)$ $f_t = \sigma(x_t W_f + b_f + s_{t-1} U_f)$ $\tilde{C}_t = \tanh(x_t W_c + b_c + s_{t-1} U_c)$ $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$ $o_t = \sigma(x_t W_o + b_o + s_{t-1} U_o)$ $s_t = o_t * \tanh(C_t)$

[a]The parameters $D = 400$ and $H = 256$ are the word embedding and LSTM internal dimensions, respectively.

[b]The parameter x_t denotes the word embedding input to the LSTM layer for the tth token in the sequence. s_t denotes the output of a node in the current layer for the current input token, and s_{t-1} denotes the output of a node in the current layer for the previous token. The input gate, i_t, weights the hidden state $\tilde{C}_t$ computed by the node for the **current** input. The forget gate f_t weights the value of the node's previous hidden state, C_{t-1}, in computing the current output. The output gate o_t weights the node's current output before passing it to the next layer. The inner activation function σ is the linearly approximated standard sigmoid, ie, $\sigma(x) = \max\left(0, \min\left(1, 0.2 * x + 0.5\right)\right)$. The operator * indicates element-wise multiplication.

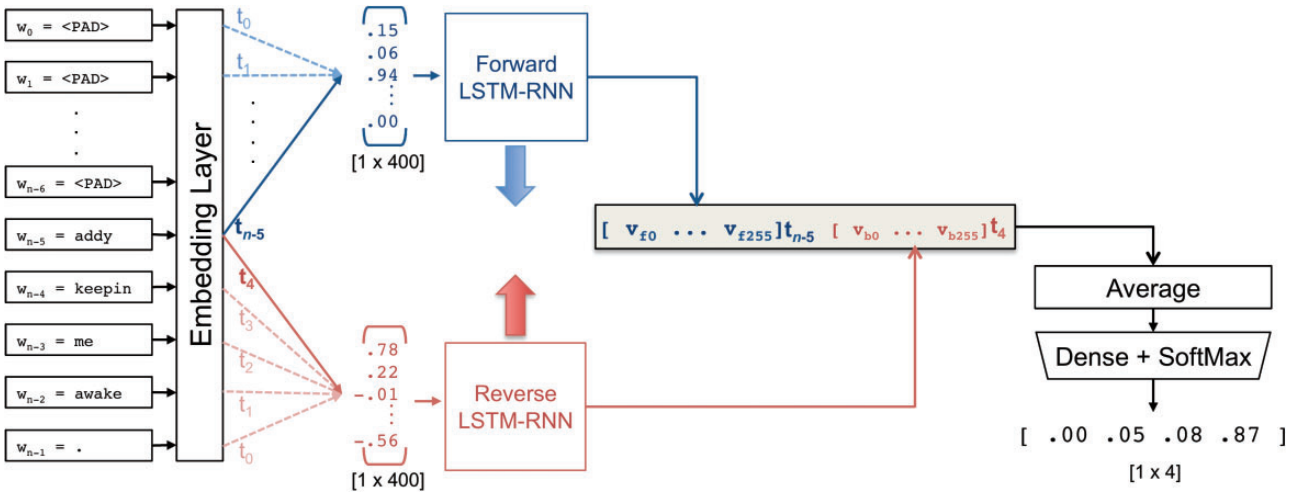

Figure 2. Overview of our bidirectional network architecture, demonstrating prediction for token w_{n-5}=addy. The network processes the entire length-n sequence in steps. At step t_i, the forward RNN predicts the label of token w_i. At step $t_{n-(i+1)}$ the reverse RNN predicts the label of token w_i. Once the entire length-n sequence is processed, the model averages the forward and reverse predictions for each token. The final fully connected layer reduces the output to 4 dimensions representing probabilities for each possible label (<PAD>, I-ADR, I-Indication, or O).

preprocessed the concept terms and tweet tokens by removing common English words (eg, "the," "of") and reducing words to their root form (eg, "jumped" to "jump"). We considered lexicon matches to constitute a *predicted* ADR if all tokens in the lexicon concept were present in the tweet, and a *true positive* approximate match if the predicted ADR at least partially overlapped with an annotated ADR span. Our additional lexicon terms and model implementation are available at https://github.com/chop-dbhi/twitter-adr-lexicon.

Conditional random fields model

Our second baseline method utilizes a CRF model[22] as described for previous state-of-the-art results.[21] We implemented the model using CRFSuite software[42] and represented each token with the following features:

- *Context Tokens*: Identities of the current token and its 3 preceding and 3 following tokens. Tokens were converted to lowercase and digits, @-mentions, and URLs were replaced with a special symbol.
- *ADR Lexicon Match*: Binary feature indicating whether the current token matches any word in the ADR lexicon.[21]
- *Part-of-Speech Tag*: The part-of-speech tag for the current token. We used the ARK Twitter Part-of-Speech Tagger.[34,43]
- *Negation*: Binary feature indicating whether the current token is negated. We utilized DepND[44–46] for negation detection.
- *Word Clusters*. The nearest word cluster for the word embeddings corresponding to the current token and the context tokens. To generate word clusters, we used the same pretrained word embeddings[28] as for the BLSTM models clustered into 100 groups using the k-means algorithm. We also considered word clusters derived from health care–specific tweets, but these did not appreciably affect F-measure (see Supplementary Appendix).

RESULTS

We used approximate matching[21,47] of predicted ADR spans on our held-out test set to evaluate model performance. Approximate matching considers a predicted ADR span correct if it overlaps with one or more actual ADR spans. Given the tweet "*The Seroquel gave me lasting sleep paralysis*" with the true ADR span "*sleep paralysis*," predicted spans of "*lasting sleep paralysis*" or simply "*paraly-*

Table 2. Mean and 95% confidence intervals for approximate match F-measure, precision, and recall achieved by each model over 10 training and evaluation rounds

Model	F-Measure	Precision	Recall
Lexicon Matching	0.6306	0.5983	0.6667
CRF	0.6507	**0.8015**	0.5477
BLSTM-M1	0.6272 (0.6123, 0.6421)	0.6457 (0.5814, 0.7101)	0.6332 (0.5645, 0.7019)
BLSTM-M2	0.6858 (0.6754, 0.6963)	0.6047 (0.5642, 0.6452)	0.8070 (0.7495, 0.8645)
BLSTM-M3	**0.7549** (0.7411, 0.7686)	0.7043 (0.6624, 0.7461)	**0.8286** (0.7608, 0.8965)

Bold text indicates the top-performing model for each metric.

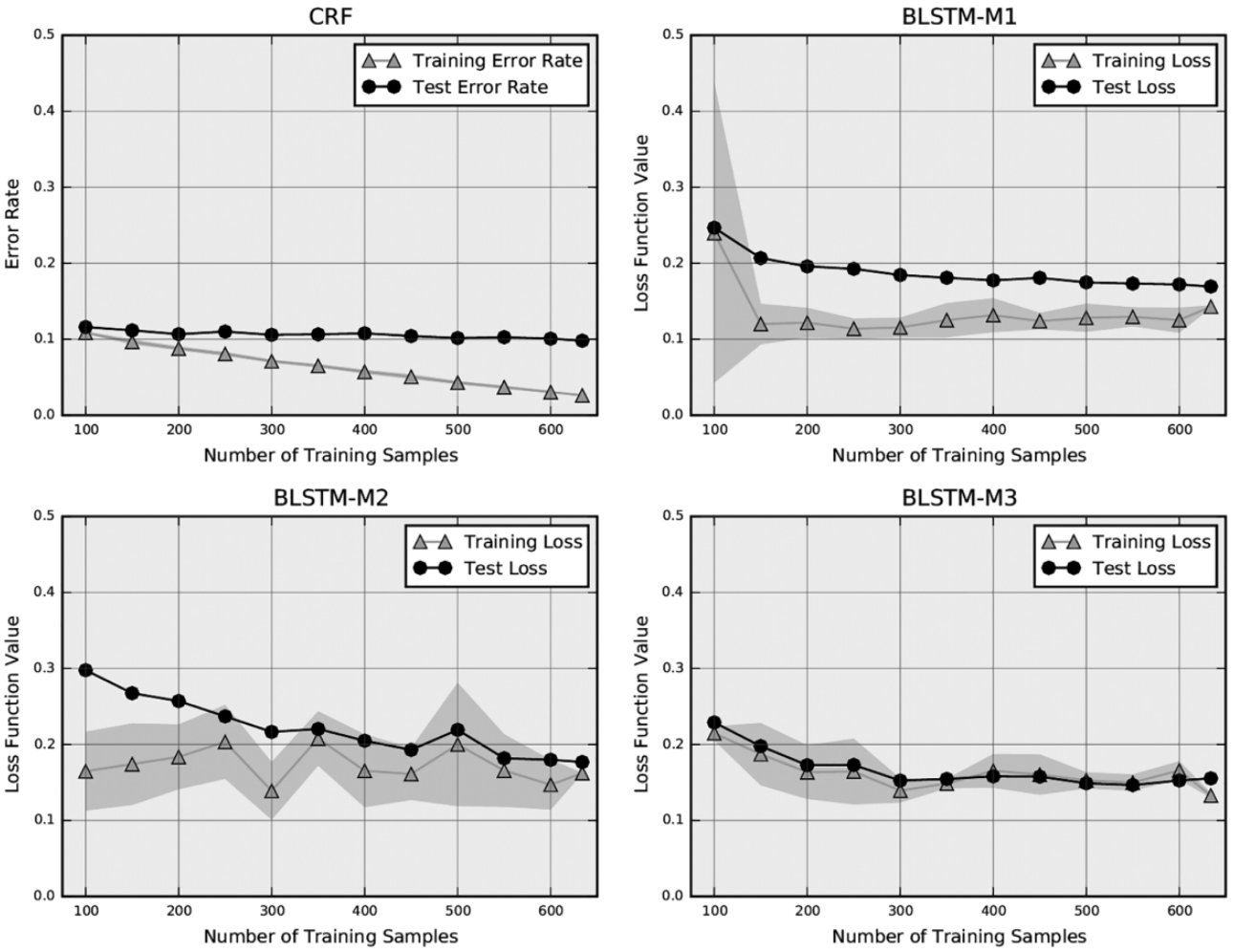

Figure 3. We observe the mean approximate match F1 score achieved by each model on the test set for training set sizes ranging from 50 to 641 samples (with 5 randomly selected training sets at each size).

sis" are counted as correct. We report the approximate match precision, recall, and F-measure:

$$\text{Precision} = \frac{\#\text{ ADR spans correctly identified}}{\#\text{ ADR spans predicted}}$$

$$\text{Recall} = \frac{\#\text{ ADR spans correctly identified}}{\#\text{ ADR spans actual}}$$

$$\text{F-Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The random order in which our model processes tweets during training can yield slightly different parameter values, thus producing slightly different predictions on the held-out test set. To account for this randomness, we trained each model on our complete training set and predicted the test set labels 10 times. We report the mean and 95% confidence interval for each metric over the 10 training and evaluation runs.

The mean approximate match results achieved by each model over the 210-tweet test set are given in Table 2. Our BLSTM-M3 model significantly outperformed all other models in terms of F-measure ($P < .01$) (statistical significance computed using the computationally intensive approximate randomization test[48–50]). Although the CRF achieved the highest precision, the BLSTM-M3

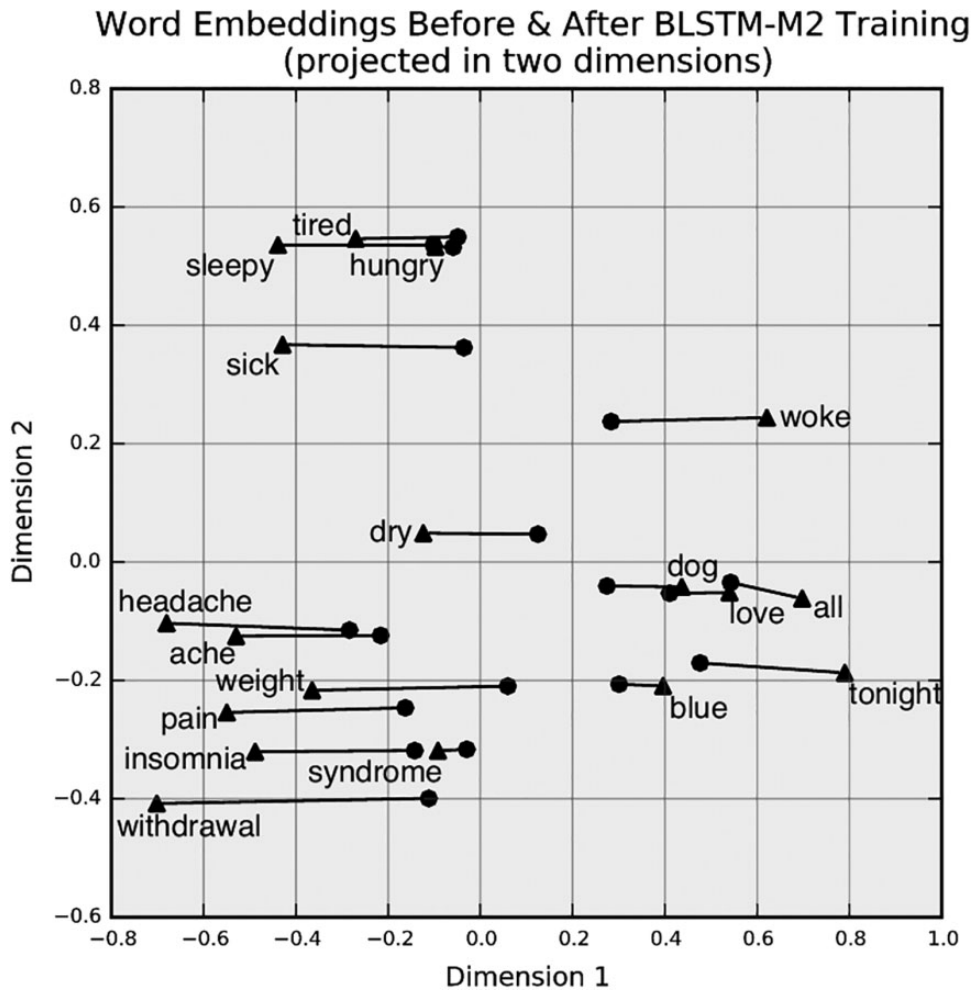

Figure 4. Word embeddings before (circles) and after (triangles) BLSTM-M2 model training, projected into 2 dimensions using singular value decomposition. Note that after training, words associated with ADR mentions (sleep, pain, weight, hungry, dreams, etc.) separate from non–ADR-associated words (blue, tonight, all, love, etc.).

model had a significantly better balance of precision and recall, as reflected in its F-measure, and thus represents new state-of-the-art performance. Both BLSTM models initialized with pretrained embeddings (BLSTM-M2 and BLSTM-M3) performed significantly better than the baselines and the BLSTM-M1 model that randomly initialized embeddings ($P < .01$). There was no statistically significant difference in performance between the Lexicon Matching, CRF, and BLSTM-M1 models ($P > .05$).

DISCUSSION

Learning curve analysis

It is important to understand a machine learning model's sensitivity to the data sample (ie, model variance), in particular to determine if the sample size is adequate. To investigate this, we examined the model learning curves (performance as function of training set size) shown in Figure 3. For the BLSTM-M3 model, the loss function values obtained during training and testing approached approximately the same asymptotic value. This typically indicates that variance is reasonably controlled and thus adding more training examples will not improve model performance. We observed that the limiting error rate for the training and test curves was non-zero (~9%), indicating that bias was present and suggesting that further performance improvements would likely require additional input features. In contrast, the CRF, BLSTM-M1, and BLSTM-M2 models' training and test error rates did not reach the same limiting values. Based on the error rate difference between the training and test evaluations, the CRF model appears to suffer the most from variance. This is likely due the greater number of input features (ie, more degrees of freedom) required by the model. Although more training examples might be beneficial for all 3 models, the downward slope of the learning curves is extremely small, and it is therefore likely that a prohibitive number of manually labeled training examples would be required to adequately control variance. Indeed, this is a strong reason to use the pretrained word embeddings, as they can be derived with unsupervised methods that do not require labeled datasets.

Feature analysis

We largely attribute the superior performance of the BLSTM-M3 model to the use of fixed word embeddings that were trained specifically to encode semantic information without specific relevance to the ADR detection task. To understand their impact, we separately evaluated model performance relative to whether the embeddings

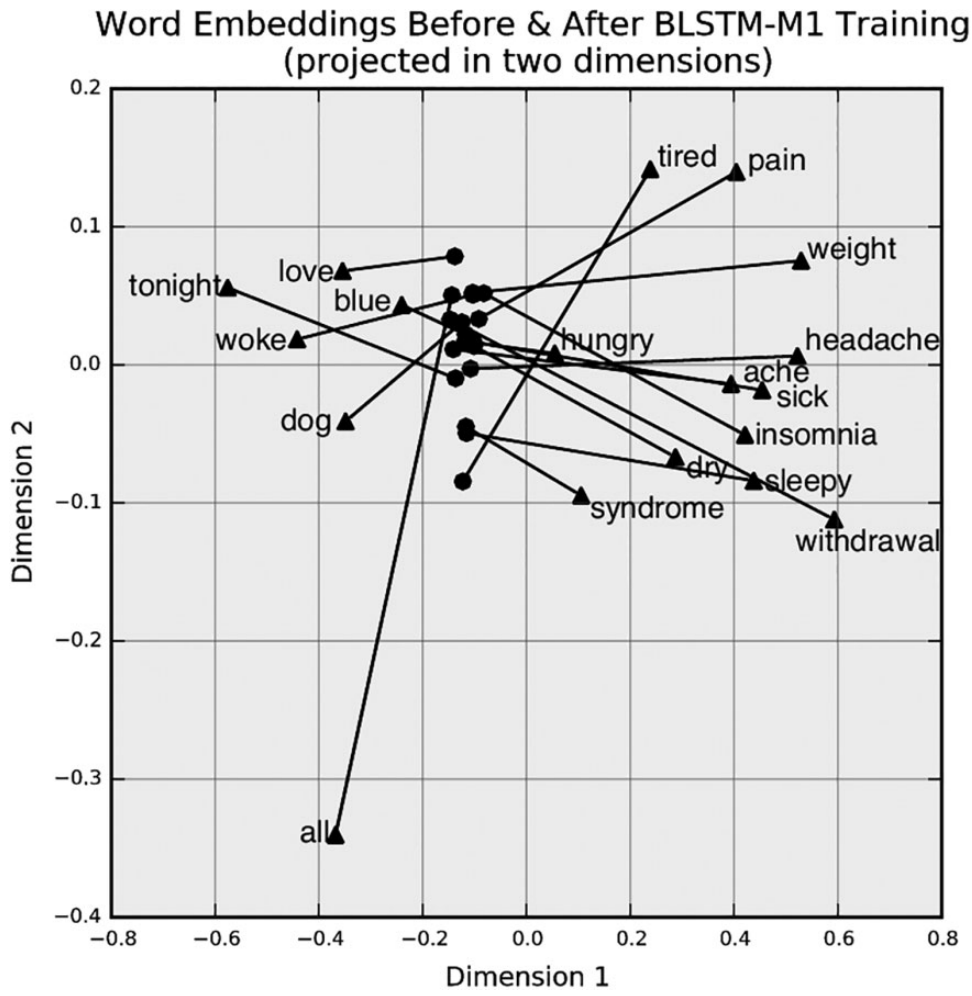

Figure 5. Projected embeddings for the same words as Figure 4, learned by our randomly initialized word embedding model, BLSTM-M1. The randomly initialized model similarly separates ADR-related and non–ADR-related words, but semantically related words are not embedded as closely as they are in the pretrained embeddings (eg, "tired" and "sleepy").

were (1) randomly initialized or pretrained, or (2) held fixed or allowed to vary during training.

We first considered the impact of using pretrained embeddings, noting that the BLSTM-M2 and BLSTM-M3 models, which did use pretrained embeddings, achieved 27.4 and 30.8% higher recall, respectively, compared to the BLSTM-M1 model, which used randomly initialized embeddings. We contend that this is a consequence of the fact that, for the pretrained embeddings, semantically similar word pairs have nearly equal embeddings, which implicitly exposes the model to information about words in the test set even if they are not observed in the training set. Consider, for example, the semantically similar ADRs "tired" and "sleepy." We expected the corresponding word vectors to be close to each other in the embedding space. If "tired" is present in the training set and "sleepy" is *only* in the test set, we can still expect the model to detect the ADR in the test set, because the RNN nodes that activate for "tired" will also activate for "sleepy." In contrast, the BLSTM-M1 model randomly initializes the embeddings, so that semantically similar word pairs are arbitrarily far apart. Such word pairs do not necessarily converge through training, because semantic similarity is not reinforced by the ADR-labeling learning objective, and even if it was, words in the test set that are not in the training set would necessarily be represented by random vectors. Thus, in our example, it would never have observed the word "sleepy" and would not recognize it as similar to "tired." In general, this results in more false negative errors (lower recall) for the randomly initialized model. Concretely, we found 553 unique words in the test set that were not in the training set, corresponding to 591 test set instances (84.6% non-ADR, 15.4% ADR) for which the BLSTM-M1, BLSTM-M2, and BLSTM-M3 models achieved false negative error rates of 13.0, 10.0, and 7.6%, respectively.

We next considered the impact of updating word embeddings relative to the ADR task, noting that the BLSTM-M3 model, which held the embeddings fixed, achieved 16.5 and 9.1% greater precision relative to the BLSTM-M2 and BLSTM-M1 models, respectively, which updated the embeddings during training. We hypothesized that this was because the BLSTM-M1 and BLSTM-M2 models encoded ADR detection information directly in the word embeddings. This is illustrated qualitatively Figures 4 and 5, where we observe that ADR-related and non–ADR-related words separate in the embedding space. This initial distinction is rigid and biases the RNN toward interpreting certain words as ADR-related or non–ADR-related, regardless of their context. Thus ADR-related words seen during training will be heavily biased toward being

Table 3. Authors' classification of false positive errors made by the BLSTM models

Classification of False Positive Error	Underlined Example of False Positive Error (Sentence annotations: **bold** = false positive; ***bold italic*** = true positive; *italic* = false negative.)	Percent of False Positive Errors (M1/M2/M3)
Constituent phrase	In other news, I've got my first weird side effect from quetiapine. *Numbness* and ***tingling*** in my **fingertips**.	9.4 / 4.8 /24.4
Other	i thought 70 mg a vyvanse was a low **dose** so i took two and now *i'm* ***bouncing off*** *the* ***walls***	34.0 / 30.6 /19.5
Indication	Fed up of **aching bones**:(Wonder drugs work some magic please. #rheumatoidarthritis #enbrel	7.5 / 4.8 /14.6
Drug benefit	whereas dextroamphetamine levels me out and makes me **feel calm** and **focused**. it's wonderful!!!!!	9.4 / 16.1 /12.2
Negated or conditional reaction	@MENTION i never had **bleeding** or **vomiting** just alot alot of *fatigue* and face was *pale* and *lost tons of* ***weight***.. #crohns #humira	5.7 / 6.5 /9.8
Symptom description	i'm **exhausted** but so ***awake*** bc vyvanse	5.7 / 8.1 /9.8
ADR-related word	@MENTION banana? Hot milk? And randomly lettuce! All contain **sleepy** bye chems. All I have is trazodone which means *dopey* all day tomo	28.3/ 29.0 /9.8

An example of each error type is underlined in the second column. Other sentence annotations are as follows: **false positive** ADR spans are in bold, ***true positive*** spans are in bold italics, and *false negative* spans are in italics.

treated as ADR-related at test time, regardless of the context, and will induce false positive errors (reduced precision). This hypothesis is supported by the data in Table 3, where we characterize the false positive errors made by each model. For the fixed embedding model (BLSTM-M3), ~10% of false positive errors are attributed to ADR-related words, ie, cases where a word appearing in a true ADR span in training is erroneously labeled as I-ADR at test time. For the variable embedding models, the share of ADR-related word errors is 3 times higher.

Error analysis

We qualitatively examined the BLSTM-M3 model errors, particularly false positives, in Table 3 to identify likely causes. Nearly 25% of the false positives can be characterized as *constituent phrase* errors, meaning that the model flags words that are part of the constituent phrase containing the true ADR but not contiguous to the annotated span. These frequently elaborate on the ADR, as in the first example in Table 3, where our model identified the word "fingertips" as part of the constituent phrase "tingling in my fingertips," but the strict ADR boundary includes only the verb "tingling." Other common errors include labeling indications, drug benefits, ADR symptoms, negated reactions, and other ADR-related words as ADRs.

Limitations

A potential limitation of our study is the size and scope of our Twitter dataset. Specifically, the limited data size could preclude generalization of our results to all drug and ADR types that might be mentioned on Twitter and other social media platforms. It is likely that there are rarely mentioned ADRs, or ADRs associated with drugs not captured by our search terms, with semantic and syntactic patterns not represented in our training set. Similarly, text from other data sources could incorporate different language patterns that require creation of additional word embeddings. The pretrained word embeddings used here embed semantic information specific to language used on Twitter. To properly evaluate our model on a data corpus with substantially different language, eg, clinical notes, requires formation of new word vectors. Fortunately, relevant datasets exist (eg, Multiparameter Intelligent Monitoring in Intensive Care [MIMIC] III[51]), and this is a focus of our continuing research.

CONCLUSION

Mirroring recent successes in other natural language processing tasks, our results indicate that RNN models with pretrained word embedding inputs can effectively identify ADRs in social media data, specifically Twitter data, and establish new state-of-the-art performance by achieving statistically significant superior F-measure performance compared to the CRF-based model.

The semantic information encoded in fixed word embeddings created from a large, non–health care–specific dataset was a critical factor in improved model performance. In particular, the semantic information improved model recall by enabling the model to recognize ADRs in the test set that were not observed in training. We also noted that model training required fewer ADR-labeled examples to reach optimal performance. This is advantageous because it reduces the required number of ADR-labeled training examples, which is important because labeling is a resource-intensive process. Additionally, word embeddings can be created using unsupervised learning methods, for which only unlabeled data are needed, and therefore can readily scale to large datasets. Finally, these embeddings are the only features used by our model, thereby avoiding task-specific feature engineering, suggesting generalizability to additional sequence-labeling tasks and simplicity over systems requiring expert-engineered input features that are difficult and expensive to develop.

While our results are encouraging, the token-labeling task is only one challenge associated with automated ADR identification. In our study, the model was trained and evaluated primarily on user posts containing drug-related events. We found that including a significant number of posts without ADRs in the training data produced relatively poor results for both the RNN and baseline models. Therefore, a fully automated system would likely require 2 additional components: (1) a highly accurate binary classifier that predicts whether text contains an ADR and (2) a method for mapping similar ADRs (eg, stomachache and stomach pain) identified through the text token labeling process to a single ontology term to support aggregated analysis. Methods for the former have been described in prior investigations.[18–20,52–54] The latter has received little attention for pharmacovigilance. Given the effectiveness of our deep learning architecture combined with the semantic information

encoded in word embeddings created from a large unlabeled dataset for ADR detection, we will explore the applicability of related methods for mapping identified ADR text to ontology terms in our future work.

FUNDING

The Leonard David Institute at the University of Pennsylvania supported this work.

COMPETING INTERESTS

AGF received an independent research grant from Pfizer, which provided salary support for his research team for work unrelated to this project.

CONTRIBUTORS

AJM and AGF conceived of and designed the study. AJM, AC, and AGF conducted data labeling and review. AJM and AC developed the model. AC implemented the models. AJM and AC analyzed the data and analyzed model performance. AJM and AC wrote the manuscript. All authors contributed to the review and revisions of the manuscript. All the authors have seen and approved the final version of the manuscript.

SUPPLEMENTARY MATERIAL

Supplementary material is available at *Journal of the American Medical Informatics Association* online.

REFERENCES

1. Hakkarainen KM, Hedna, K, Petzold M, *et al*. Percentage of patients with preventable adverse drug reactions and preventability of adverse drug reactions: a meta-analysis. *PLoS One*. 2012;7(3):e33236.
2. Sultana J, Cutroneo P, Trifiro G. Clinical and economic burden of adverse drug reactions. *J Pharmacol Pharmacother*. 2013;4:S73–77.
3. Ahmad, SR. Adverse drug event monitoring at the Food and Drug Administration. *J Gen Intern Med*. 2003;18(1):57–60.
4. Li H, Guo, XJ, Ye XF, *et al*. Adverse drug reactions of spontaneous reports in Shanghai pediatric population. *PLoS One*. 2014;9(2):e89829.
5. Lindquist M. VigiBase, the WHO global ICSR database system: basic facts. *Drug Inform J*. 2008;42(5):409–19.
6. Behrman RE, Benner JS, Brown JS, *et al*. Developing the Sentinel System: a national resource for evidence development. *N Engl J Med*. 2011;364(6):498–99.
7. Reisinger SJ, Ryan PB, O'Hara DJ, *et al*. Development and evaluation of a common data model enabling active drug safety surveillance using disparate healthcare databases. *J Am Med Inform Assoc*. 2010;17(6):652–62.
8. Nadkarni PM. Drug safety surveillance using de-identified EMR and claims data: issues and challenges. *J Am Med Inform Assoc*. 2010;17(6):671–74.
9. Wang X, Hripcsak G, Markatou M, *et al*. Active computerized pharmacovigilance using natural language processing, statistics, and electronic health records: a feasibility study. *J Am Med Inform Assoc*. 2009;16(3):328–37.
10. LePendu P, Iyer SV, Bauer-Mehren A, *et al*. Pharmacovigilance using clinical notes. *Clin Pharmacol Ther*. 2013;93(6):547–55.
11. Harpaz R, Vilar S, Dumouchel W, *et al*. Combing signals from spontaneous reports and electronic health records for detection of adverse drug reactions. *J Am Med Inform Assoc*. 2013;20(3):413–19.
12. Leaman R, Wojtulewicz L, Sullivan R, *et al*. Towards internet-age pharmacovigilance: extracting adverse drug reactions from user posts to health-related social networks. In: *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*; 2010:117–25.
13. Benton A, Ungar L, Hill S, *et al*. Identifying potential adverse effects using the web: a new approach to medical hypothesis generation. *J Biomed Inform*. 2011;44(6):989–96.
14. Yang C, Jiang L, Yang H, *et al*. Detecting signals of adverse drug reactions from health consumer contributed content in social media. In: *Proceedings of the 18th Association for Computing Machinery Special Interest Group on Knowledge Discovery and Data Mining Workshop on Health Informatics (HI-SIGKDD)*; 2012:1–8.
15. Yates A, Goharian N. ADRTrace: detecting expected and unexpected adverse drug reactions from user reviews on social media sites. *Adv Inform Retrieval*. 2013;7814LNCS:816–19.
16. White RW, Tatonetti NP, Shah NH, *et al*. Web-scale pharmacovigilance: listening to signals from the crowd. *J Am Med Inform Assoc*. 2013;20(3):404–08.
17. Freifeld CC, Brownstein JS, Menone CM, *et al*. Digital drug safety surveillance: monitoring pharmaceutical products in Twitter. *Drug Saf*. 2014;37(5):343–50.
18. Ginn R, Pimpalkhute P, Nikfarjam A, *et al*. Mining Twitter for adverse drug reaction mentions: a corpus and classification benchmark. In: *Proceedings of the Fourth Workshop on Building and Evaluating Resources for Health and Biomedical Text Processing*; 2014:1–8.
19. Liu, X, Liu J, Chen H. Identifying adverse drug events from health social media: a case study on heart disease discussion forums. In: *Proceedings of the International Conference on Smart Health (ICSH)*. 2014;8549:25–36.
20. O'Connor K, Nikfarjam A, Ginn R, *et al*. Pharmacovigilance on Twitter? Mining tweets for adverse drug reactions. *AMIA Annu Symp Proc*; 2104:924–33.
21. Nikfarjam A, Sarker A, O'Connor K, *et al*. Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features. *J Am Med Inform Assoc*. 2015;22(3):671–81.
22. Lafferty J, McCallum A, Pereira F. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: *Proceedings of the 18th International Conference on Machine Learning*. 2001;1:282–89.
23. Wang, W. Mining adverse drug reaction mentions in Twitter with word embeddings. In: *Online Proceedings of the Pacific Symposium on Biocomputing Social Media Mining Shared Task Workshop 2016*. http://diego.asu.edu/psb2016/acceptedpapers/DLIR.pdf. Accessed August 1, 2016.
24. Sarker, A, Nikfarjam, A, Gonzalez, G. Online Proceedings of the Social Media Mining Shared Task Workshop. *Pacific Symposium on Biocomputing*. 2016;21:581–92.
25. Yao K, Zweig G, Hwang MY, *et al*. Recurrent neural networks for language understanding. In: *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH)*; 2013:2524–2528.
26. Graves A. Supervised sequence labeling with recurrent neural networks (doctoral dissertation). *Studies in Computational Intelligence 385*, Springer; 2012:1–131.
27. Mesnil G, He X, Deng L, *et al*. Investigation of recurrent-neural-network architectures and learning methods for language understanding. In: *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2013:3771–75.
28. Godin F, Vandersmissen B, De Neve W, *et al*. Multimedia Lab @ ACL W-NUT NER Shared Task: named entity recognition for Twitter microposts using distributed word representations. In: *Proceedings of the Association for Computational Linguistics 2015 Workshop on Noisy User-generated Text*; 2015:146–53.
29. Lipton ZC, Kale DC, Elkan C, *et al*. Learning to diagnose with LSTM recurrent neural networks. In: *Proceedings of the International Conference on Learning Representations*. 2016. arxiv.org/pdf/1511.03677v6.pdf. Accessed March 2, 2016.

30. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9(8):1735–80.
31. Schuster M, Paliwal KK. Bidirectional recurrent neural networks. *IEEE Trans Audio Speech Lang Process*. 1997;45(11):2673–81.
32. Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw*. 2005;18(5–6):602–10.
33. Ott M. ark-twokenize-py. *GitHub Repository*. 2016. github.com/myleott/ark-twokenize-py. Accessed August 1, 2016.
34. Owoputi O, O'Connor B, Dyer C, *et al*. Part-of-speech tagging for Twitter: word clusters and other advances. Carnegie Mellon University. CMU-ML-12-107. 2012. www.cs.cmu.edu/~ark/TweetNLP/owoputi+etal.tr12.pdf. Accessed August 2016.
35. Ramshaw LA, Marcus MP. Text chunking using transformation-based learning. In: *Proceedings of the Association for Computational Linguistics Third Workshop on Very Large Corpora*. 1995:82–94.
36. Mikolov T, Chen K, Corrado G, *et al*. Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781. 2013. arxiv.org/pdf/1301.3781v3.pdf. Accessed August 1, 2016.
37. Chollet F. Keras. *GitHub Repository*. 2016. github.com/fchollet/keras.. Accessed August 1, 2016.
38. Bergstra J, Breuleux O, Bastien F, *et al*. Theano: a CPU and GPU math expression compiler. In: *Proceedings of the 9th Python for Scientific Computing Conference (SciPy)*. 2010:1–7.
39. Bastien F, Lamblin P, Pascanu R, *et al*. Theano: new features and speed improvements. In: *Adv Neural Inf Process Syst Workshop on Deep Learning and Unsupervised Feature Learning*; 2012:1–10.
40. Werbos, PJ. Backpropagation through time: what it does and how to do it. *Proc IEEE*. 1990;78(10):1550–60.
41. *Open Source Collaborative Consumer Health Vocabulary Initiative*. consumerhealthvocab.org. Accessed August 1, 2016.
42. Okazaki N. CRFsuite: A Fast Implementation of Conditional Random Fields. Software Package. 2007. www.chokkan.org/software/crfsuite/. Accessed June 1, 2016.
43. Gimpel K, Schneider N, O'Connor B, *et al*. Smith. Part-of-speech tagging for Twitter: annotation, features, and experiments. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics, Companion Volume*. Portland, OR; June 2011.
44. Guo Z. DepND. *GitHub Repository*. 2016. github.com/zachguo/DepND. Accessed August 1, 2016.
45. Sagae, K *GDep (GENIA dependency parser)*. Software package. 2016. sagae.bitbucket.org/gdep/. Accessed August 1, 2016.
46. Sagae K, Tsujii J. Dependency parsing and domain adaptation with LR models and parser ensembles. In: *Proceedings of the 11th Conference on Computational Natural Language Learning*; 2007:1044–50.
47. Tsai RT, Wu SH, Chou WC, *et al*. Various criteria in the evaluation of biomedical named entity recognition. *BMC Bioinformatics*. 2006;7:92–100.
48. Noreen EW. Approximate randomization tests. In: EW Noreen, ed. *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Hoboken, NJ: John Wiley & Sons, Inc; 1989:9–33.
49. Cohen P. *Empirical Methods for Artificial Intelligence*. Cambridge, MA: MIT Press; 1995:165–75.
50. Padó, S. User's guide to sigf: Significance Testing by Approximate Randomization. 2006. http://www.nlpado.de/~sebastian/software/sigf.shtml. Accessed August 2016.
51. Johnson AEW, Pollard TJ, Shen L, *et al*. MIMIC-III, a freely accessible critical care database. *Scientific Data*. 2016; 3: 160035.
52. Patki A, Sarker A, Pimpalkhute P, *et al*. Mining adverse drug reaction signals from social media: going beyond extraction. In: *Proceedings of the BioLINK Special Interest Group*. 2014:1–8.
53. Sarker A, Gonzalez G. Portable automatic text classification for adverse drug reaction detection via multi-corpus training. *J Biomed Inform*. 2015; 53: 196–207.
54. Korkontzelos I, Nikfarjam A, Shardlow M, *et al*. Analysis of the effect of sentiment analysis on extracting adverse drug reactions from tweets and forum posts. *J Biomed Inform*. 2016; 62: 148–58.