Published in final edited form as:

IEEE Trans Nanobioscience. 2019 July ; 18(3): 324–334. doi:10.1109/TNB.2019.2909094.

Serendipity—A Machine-Learning Application for Mining Serendipitous Drug Usage from Social Media

Boshu Ru,
Department of Software and Information Systems University of North Carolina at Charlotte Charlotte, NC, USA

Dingcheng Li,
Big Data Lab Baidu USA, Inc Bellevue, WA, USA

Yueqi Hu,
Department of Computer Science University of North Carolina at Charlotte Charlotte, NC, USA

Lixia Yao
Department of Health Sciences Research Mayo Clinic Rochester, MN, USA

Abstract

Serendipitous drug usage refers to the unexpected relief of comorbid diseases or symptoms when taking a medication for a different known indication. Historically, serendipity has contributed significantly to identifying many new drug indications. If patient-reported serendipitous drug usage in social media could be computationally identified, it could help generate and validate drug-repositioning hypotheses. We investigated deep neural network models for mining serendipitous drug usage from social media. We used the word2vec algorithm to construct word-embedding features from drug reviews posted in a WebMD patient forum. We adapted and redesigned the convolutional neural network, long short-term memory network, and convolutional long short-term memory network by adding contextual information extracted from drug-review posts, information-filtering tools, medical ontology, and medical knowledge. We trained, tuned, and evaluated our models with a gold-standard dataset of 15,714 sentences (447 [2.8%] describing serendipitous drug usage). Additionally, we compared our deep neural networks to support vector machine, random forest, and AdaBoost.M1 algorithms. Context information helped reduce the false-positive rate of deep neural network models. If we used an extremely imbalanced dataset with limited instances of serendipitous drug usage, deep neural network models did not outperform other machine-learning models with n-gram and context features. However, deep neural network models could more effectively use word embedding in feature construction, an advantage that makes them worthy of further investigation. Finally, we implemented natural-language processing and machine-learning methods in a web-based application to help scientists and software developers mine social media for serendipitous drug usage.

Keywords

social media; health informatics; data mining; drug discovery; drug repurposing

[*]Contact of corresponding author: yao.lixia@mayo.edu.

I. Introduction

Drug repositioning is the identification of novel indications for marketed drugs and for drugs in the late stages of development [1]. A well-known example of drug repositioning is sildenafil, which was originally developed to treat angina and was repurposed to treat erectile dysfunction [2]. Repositioned drugs have a better safety profile than compounds in the early stages of discovery and development because they have already passed several preclinical tests. Therefore, some time and cost associated with preclinical testing can be saved, making repositioned drugs more readily available to patients with improperly treated diseases, as well as more cost-efficient for pharmaceutical companies [3]. These advantages are of interest to biomedical researchers, who have examined various computational methods to generate and verify drug-repositioning hypotheses by assessing chemical and biological data, literature, and electronic health records [1, 4–10].

In the past decade, fast-growing social media websites have reached a critical mass of patient discussions about diseases and drugs [11], primarily in the form of unstructured, casual human language. These data cover various medication outcomes such as effectiveness, adverse effects due to medication, adherence, and cost [11]. Recent research has examined this new data source primarily for pharmacovigilance purposes [12–17]. In social media posts, some patients have also mentioned that comorbid diseases or symptoms unexpectedly improved while they were taking a certain drug for a common or known indication. We refer to these events as *serendipitous drug usage*. Fig. 1 shows an example of serendipitous drug usage: a patient reported that her symptoms of irritable bowel syndrome were alleviated when taking sulfasalazine, which was prescribed for rheumatoid arthritis. Such information could be helpful for generating and verifying drug-repositioning hypotheses if these statements could be computationally detected from the overwhelming amount of noise in social media data.

In previous work[18], we explored natural language processing (NLP) and machine-learning methods to identify serendipitous drug usage from patient health forums (social media). We collected drug-review posts from WebMD and designed information filters to eliminate noise in the data. We developed the first gold-standard dataset for predicting serendipitous drug usage; it consisted of 447 sentences from WebMD that mentioned serendipitous drug usage and 15,267 sentences that did not. Then, we constructed machine-learning features from n-grams, outputs from information-filtering tools, medical knowledge, and other information from the drug-review posts. We used machine-learning algorithms, namely support vector machine (SVM), random forest, and AdaBoost.M1, to detect serendipitous drug usage. Our best model had an area under the receiver operating characteristic curve (AUC) of 0.937, precision of 0.811, and recall of 0.476. Several of our predictions, including metformin and bupropion for obesity, tramadol for depression, and ondansetron for irritable bowel syndrome with diarrhea, were also supported by recent biomedical research publications.

Deep neural networks such as the convolutional neural network (CNN), long short-term memory network (LSTM), and convolutional LSTM (CLSTM) recently have demonstrated superb performance in text-classification tasks [19–23]. For example, Zhou et al. [20] designed CLSTM to predict the sentiment of sentences and achieved an accuracy of 0.878 on the Stanford Sentiment Tree Bank dataset. Jia et al. [23] ensembled multiple CNNs to automatically assign products to 1 of 3,008 categories and reported a weighted F1 score of 0.8295. These studies typically used word embedding that was trained by unsupervised learning algorithms such as word2vec [24] to construct features from texts, and these features were then classified by using convolutional filters or recurrently connected neurons. Several papers reported that deep neural networks outperformed traditional machine-learning algorithms such as SVM, logistic regression, and random forest on a number of well-known annotated text corpora [20, 25]. These promising results motivated us to investigate deep neural networks for detecting serendipitous drug usage in social media. If successful, these efforts would potentially accelerate drug discovery and development, particularly for therapeutic areas with insufficient financial investment.

For this study, we trained a word embedding from 800,000 drug-review sentences that we collected from WebMD and used it to construct features for the 15,714 sentences in the gold-standard dataset that we previously developed [18]. We redesigned the well-known deep neural networks of CNN by Kim [19], LSTM [26], and CLSTM [20] by adding a fully connected neural network (FCN) to predict serendipitous drug usage by considering social media text and contextual information. We used cost-sensitive learning to reduce the impact of imbalanced data. We compared original and redesigned deep neural networks to SVM, random forest, and AdaBoost.M1 algorithms. Additionally, we developed *Serendipity*, an easy-to-use, web-based software application, to effectively extract serendipitous drug usage from social media. Unlike the computational pipeline that we used in the current study and our previous research, this web application takes only drug-review comments and the drug name as inputs, making it adaptive to broader sources of patient-generated health data. The NLP and machine-learning methods are integrated into an automated workflow. *Serendipity* provides a graphic user interface (GUI) for scientists working in drug discovery and development who have limited programming experience, and it also has an application programming interface (API) for software developers who want to integrate *Serendipity* into other programs.

II. Method

A. Data

We used the gold-standard dataset from our previous study [18] to build and evaluate our deep neural network models. This dataset contains 15,714 sentences from WebMD, including 447 that mention serendipitous drug usage. In addition to drug-review sentences and annotation labels, the dataset also includes 169 context information fields (Table I) that were constructed from WebMD, natural language processing (NLP) tools (MetaMap [32], Stanford CoreNLP [33]), clinical ontology (SNOMED CT [34]), and our knowledge of drug therapeutic areas where repositioning opportunities often arise.

B. Word embedding

A word embedding represents words as dense vectors in a high-dimensional vector space (usually 50–300 dimensions). Recent studies have trained neural networks with a large, unannotated text corpus to construct word embeddings [24, 30]. In the vector space of such word embeddings, words with syntax and semantic relations tend to be close to each other [24, 31].

We began training a word embedding from 800,000 drug-review sentences that we collected from WebMD; the word embedding was trained with the word2vec algorithm [24] from the Gensim Python library [32]. Sentences were preprocessed to remove non-English characters, convert letters to lower case, and stem words to their basic form. We set the dimension of word vectors d to be 200 and specified the context window size to be 50 because more than 99% of sentences in the WebMD corpus are shorter than 50 words. The resulting WebMD word embedding contained 67,659 word vectors, and the dimension of each vector was equal to 200.

We then used the WebMD word embedding to construct machine-learning features for sentences in the gold-standard dataset. These features are commonly constructed by 3 methods. The first method sums vectors for each word in a sentence and optionally divides the aggregation results by the number of words in the sentence [33]. The second method uses clustering [17], which groups (clusters) all words in the word embedding with algorithms such as K-means and then represents each sentence as a vector, with values indicating the distribution of words across clusters. Both methods transform word vectors of each sentence into a set with a fixed number of attributes that can then be the input features for machine-learning models such as SVM and random forest. A third method, however, concatenates word vectors to a sentence vector. For a sentence of i words $\omega_1, w_2, \ldots, \omega_i$, it concatenates word vectors $\upsilon_{\omega 1}, v_{\omega 1}, \ldots, \upsilon_{\omega i}$, each with d dimensions, to form a concatenated feature vector of dimension $i \times d$. This method is not immediately applicable to machine-learning models such as SVM and random forest because the location of each word varies across different sentences and so as the features. However, such feature vectors keep information about word sequence [33], which can be captured by deep neural networks [19, 20] using special filters (convolution and max-pooling filters in CNN, information gates in LSTM; see section II-C). We used the vector concatenation approach and set feature vectors to be 50 words × 200 dimensions by padding zeros to shorter sentences and trimming excessive words from longer sentences.

C. Deep Neural Network Models

CNN: Several recent studies examined CNN-based neural network models and showed outstanding performance in text-classification tasks [19, 22, 23]. We began with Kim's CNN model [19] and used our WebMD word-embedding features. The model (Fig. 2) transformed the input sentence to a matrix by using the concatenation approach (see section II-B) and contained parallel convolution filters of 3 different sizes ($k-1$, k, and $k+1$ with n_c filters), max-pooling filters, and a fully connected layer of neurons for prediction. The convolution filters were trained to extract informative patterns from a subarea of the input data. The kernel size (*k*) of the filter determined the magnitude of the subarea. In the case of text, *k* is the number of continuous words in a sentence. By mixing convolution filters of 3 continuous sizes ($k-1$, k, and $k+1$), the network can learn patterns in the sentence at 3 different scales (the Hyperparameter Tuning section describes how we selected *k*; see section II-D). This design is similar to combining n-gram features of different scales (e.g., unigram, bigram, and trigram).

The max-pooling filter moves a “window” over outputs of convolution filters and preserves only maximum values in the current scanning area. The dimension of the scanning area was determined by the pooling-window size parameter. This operation eliminates information that is less relevant to the classification task, such as random patterns and blank (padding) areas, and reduces the dimensionality of features extracted by convolution filters. The output was further processed by 1 layer of fully connected neurons (dense 0 in Fig. 2) before it was submitted to a single neuron for prediction.

Additionally, we added dropout layers (dropout 0 and dropout 1 in Fig. 2) and an *l2* kernel regularizer [34] to prevent the CNN model from overfitting. The dropout layers randomly intercepted output of previous neurons in the training process and the *l2* kernel regularizer penalized large-magnitude weights while fitting the network. These 2 methods were also used in other deep neural network models.

LSTM: Although convolution filters are good at processing data in the matrix or grid representation, they capture only sequential patterns in a local area and sometimes miss long-range dependencies between words in the same sentence [35]. The LSTM network, a special type of recurrent neural network that uses LSTM units, was introduced to solve this problem [36]. While processing sequential data, each LSTM unit leverages 4 information gates to decide which new and existing information should be added or removed from the information flow [36]. Our second model used a 1-directional LSTM network along with 2 dropout layers and 1 layer of fully connected neurons to process drug-review posts (Fig. 3).

CLSTM: The third model adapted the approach of Zhou et al. [20], which added LSTM to the CNN (Fig. 4). This design leverages convolution and max-pooling filters to extract the most discriminative patterns from local areas of the word-embedding feature matrix and continuously feeds the signals to LSTM, which focuses on learning sequential patterns in the information flow. The rest of the CLSTM model is similar to the CNN model.

Deep neural networks with context information features: Drug-review comments in social media often include context information fields that describe the patient, disease, and drug. Such information can be used with medical knowledge and NLP methods to enrich the context of social media data. For example, our gold-standard dataset integrated drug-review sentences with the patient’s basic demographic information, ratings for the drug, drug therapeutic areas, and outputs from the filtering tools (Table I). To combine social media text and context information for making predictions, we designed new models (Fig. 5) with a deep neural network and an FCN. For text features, we used the architecture of the original CNN, LSTM, and CLSTM networks but removed the prediction layer. For context information features, we designed a neural network containing 3 layers of fully connected neurons (dense 1–3). Each layer had half the neurons of the previous layer to condense the output. We then combined outputs from text features and context information features together to an additional layer of fully connected neurons (dense 4) and another dropout layer (dropout 2) before making the prediction. We named these new models as CNN+FCN, LSTM+FCN, and CLSTM+FCN.

D. Model Implementation

Platform: We implemented the original and redesigned deep neural network models in an Ubuntu 16 Linux system with Python 3.6.2, Keras 2.0.8 (a well-known deep neural network library for Python) [37], and TensorFlow 1.3.0 math library [38] for high-efficiency neural network computing.

Model configuration: We chose the rectified linear unit [39] as the activation function for all neural network nodes, except the single neuron for prediction. The rectified linear unit computes the function of $relu(x)=max(x,0)$, where the input variable x is a weighted sum of real value numbers from previous neurons in the network. It has been a popular approach in recent deep neural network research because it does not saturate (the gradients do not diminish when x approaches positive infinity) and is less computationally expensive than functions that involve the exponential calculation. For the prediction layer, we chose the sigmoid function $sigmoid(x)=1/(1+e^{-x})$, which projects a weighted sum of previous neurons to a range from 0 to 1 [40]. We selected Adam as the kernel optimizer [41] and binary cross-entropy as the loss function [42].

Data preprocessing: Data preprocessing: All numeric features were linearly rescaled to the range of [−1, 1], and all categorical features were converted to binary vectors. We then split the 15,714 annotated sentences by posting date into training, validation, and test datasets. Of the sentences, 60% (9,429 sentences) were posted from September 18, 2007, through December 7, 2010, and were used as the training dataset to fit deep neural networks. Next, 20% (3,142 sentences) that were posted from December 8, 2010, through October 11, 2012, were used as the validation dataset to tune hyperparameters of the networks. The remaining 20% (3,143 sentences) were posted from October 12, 2012, through March 26, 2015, and were used as the independent test dataset. In the 3 datasets, the proportion of serendipitous drug usage ranged from 2.0% to 3.2%.

Hyperparameter tuning: We tuned hyperparameters, including the kernel size (k) and number of convolution filters (n_c), the size of the pooling window for max-pooling filters, the number of neurons for each dense layer, the drop ratio for each dropout layer, the constant parameter for the $l2$ kernel regularizer [34] that was applied to the prediction neuron, and the number of units in the LSTM network. We also searched for the best method to initialize the weights of the neural network among 6 commonly used initializers: random uniform, random normal, Xavier uniform, Xavier normal, He uniform, and He normal [40, 43]. Moreover, neural networks are sensitive to imbalanced data [44]. Keras provides a cost-sensitive learning solution by allowing us to specify the importance of each class while fitting the neural network. To fully leverage this feature, we treated class weights as an additional tunable hyperparameter.

We fit models on the training dataset with different sets of hyperparameters and tracked each model’s performance on the validation dataset. We used Hyperas [45], a parameter-tuning library for Keras. Unlike grid search, Hyperas does not exhaustively search the entire hyperparameter space. Instead, it leverages search algorithms such as tree of Parzen estimators [46] to partially search the parameter space for a relatively good parameter setting. Hyperas has been widely used in recent deep neural network research and applications because the search spaces for hyperparameters are often too big to complete a grid search in a manageable amount of time.

Evaluation: We evaluated 6 deep neural network models on the test dataset, namely CNN, LSTM, CLSTM, CNN+FCN, LSTM+FCN, and CLSTM+FCN, in terms of AUC, precision, and recall [47]. Additionally, we compared deep neural network models to models built from 3 widely used machine-learning algorithms, namely SVM, random forest, and AdaBoost.M1. For each nonneural network algorithm, we built 2 models, one with n-gram and context features [18] and the other with word-embedding features and context features.

E. Web Application Design and Implementation

We implemented *Serendipity* by using Flask, a lightweight Python framework for web development [48]. The implementation also depended on third-party tools and libraries, including Stanford CoreNLP [28], MetaMap [27], Scikit-learn [49], Keras [37], and several commonly used JavaScript libraries [50–52]. The system architecture followed the design pattern of Model-View-Controller [53]. The view layer provides 2 user interfaces. One is a GUI written in HTML5 [54] and the other is a RESTful API [55]. The model layer conducts NLP and machine-learning tasks to extract serendipitous drug usage. The controller layer coordinates information flow between the view and model layers and visualizes serendipitous drug usage mining results. The major components and workflow of the system are shown in Fig. 6.

User interfaces: We recognized that scientists with knowledge in drug discovery and development but limited computer programming skills are potential users of this software. For these users, we built a GUI with HTML5, a webpage markup language that is compatible with most recent web browsers on computers, smart phones, and tablets [54]. The GUI (Fig. 7) has 2 input fields, one for social media text and the other for the drug associated with the text. When the user types in the drug name, the GUI automatically shows a list of drugs to choose. These are drugs in the databases that match the input. Then, the user can submit the social media text for analysis (select the “Go” button) or clear all inputs (select the “Reset” button). After submission, the model layer will mine serendipitous drug usage and the controller layer will generate a new interactive webpage for visualizing the results.

Other potential users of *Serendipity* are software developers with proficiency in computer programming but limited experience with NLP and machine-learning tools. For users who want to integrate *Serendipity* into their own software applications, we provided a RESTful API [55] that accepts social media text and drug ID from the user as HTTP requests (Fig. 8). The drug ID are assigned by our system, and a separate file shows the mapping between the identification number and drug name. After the model layer finishes NLP and machine-learning prediction steps, the RESTful API returns results as a JavaScript Object Notation (JSON) object [56], which is a commonly used structured format for exchanging data between software applications.

The model layer: Our system used SQLite [57] database to store common usages (known indications) and therapeutic area information for 1,963 drugs that we assessed in a previous study [18]. The common usages were saved in plain text and by using the corresponding concept unique identifiers (CUIs) in the Unified Medical Language System (UMLS) [58]. The model layer used Stanford CoreNLP to split social media text into sentences and assign a sentiment score (very negative, negative, neutral, positive, or very positive) [59] to each sentence. Then, sentences were processed by MetaMap to map diseases and symptoms mentioned in the sentence to the CUI in UMLS ($\mathrm{CUI}_{\text{social media drug usage}}$ in Fig. 6). To quantify the difference between diseases and symptoms mentioned in the social media text and those of known drug indications, the model layer retrieved UMLS concepts associated with known indications of the drug ($\mathrm{CUI}_{\text{known indications}}$ in Figure 6) from the SQLite database. Next, it calculated the semantic difference between $\mathrm{CUI}_{\text{social media drug usage}}$ and $\mathrm{CUI}_{\text{known indications}}$ based on the distance between them in a directed acyclic graph constructed from SNOMED CT [18, 29]. In addition to the sentiment and semantic difference analyses, the model layer also extracted the n-grams (n=1, 2, and 3) and concatenated word-embedding vectors from each sentence, using the method described in [18] and section II-B. Finally, features extracted from the drug information database, sentiment analysis, semantic difference calculation, and social media text were passed through 6 machine-learning models to predict the probability that serendipitous drug usage would be mentioned in the sentence. These models included traditional models (SVM, random forest, and AdaBoost.M1) that we explored previously [18] and the new models CNN+FCN, LSTM+FCN, and CLSTM+FCN described above (see section II-C).

Visualization: The GUI (Fig. 9) can display 9 types of information that might be of interest to scientists working in drug discovery and development. The name and common usages (known indications) of the drug in the SQLite database are presented at the top. The lower area is divided into left and right panels. The UMLS-preferred disease names and symptoms mentioned in the social media text are listed as clickable tags in the left panel; tags are presented in descending order of their average *Serendipity* score, as determined by prediction models. Each tag can be expanded (by clicking) to display the CUI, semantic type, trigger word (the original word in the sentence that was mapped to the UMLS concept), and the average *Serendipity* score from the prediction models. To limit the number of tags in the left panel, disease and symptom tags are excluded if they meet any of the following criteria: 1) the part of speech for the trigger word is not a noun; 2) the average model prediction score is less than 3%; 3) the semantic difference to common usages of the drug is less than 3 steps in the SNOMED CT graph; and 4) the sentiment is not positive. The latter 2 conditions reflect the nature of serendipitous drug usage. The usage cannot be the same as or too similar to known indications. The sentiment cannot be negative because negative feelings are often associated with adverse drug effects. The right panel displays the social media text verbatim, sentence by sentence. Red dots appear under a sentence if the sentiment is negative or very negative. Green dots appear if the sentiment of the sentence is positive or very positive. All trigger words in the right panel are highlighted in yellow. When the user clicks a tag in the left panel, the right panel will automatically scroll to the location of corresponding sentence, and the trigger word will be highlighted in red.

RESTful API output: For software developers, the RESTful API returns NLP and machine-learning results as a JSON object. Because these users may have diverse ideas about how they to use these results, we constructed the JSON object to include 19 data elements; Table II shows the description and source for each element. Additionally, the results are not filtered.

A prototype of the web application is accessible at https://github.com/boshuru/serendipity for demonstration.

III. Results and Discussion

A. Hyper parameters

Using Hyperas and the validation dataset, we tuned hyperparameters for each model (Table III). The kernel sizes of convolution filters were higher for CNN and CNN+FCN than for CLSTM and CLSTM+FCN. Convolution filters were used primarily to extract informative patterns from local areas of a matrix. The kernel size determined the scale of the local area, which was defined as the continuous number of words in a sentence. In the WebMD dataset, 75% of sentences contained fewer than 19 words and 50% contained fewer than 13 words. For CNN and CNN+FCN models, the optimal kernel sizes is relatively large (7 to 13 words) to cover a sufficient number of words in each sentence. After adding LSTM to the CNN, the optimal convolution kernel sizes ranged from 3 to 7. With LSTM learning sequential patterns between words, CNN could focus on patterns in smaller areas (3 to 7 words).

We next determined how much the hyperparameter tuning affected the performance of deep neural networks. Table IV lists the minimum, average, and maximum values for AUC, precision, and recall on the validation dataset for various hyperparameter sets. The spread between the minimum and maximum was wide for all machine-learning performance metrics and deep neural networks, highlighting the importance of tuning hyperparameters.

B. Model evaluation

We evaluated deep neural network models in terms of AUC, precision, and recall on the test dataset (Table V) and set true serendipitous drug usage as the positive class. Among deep neural network models, the highest AUC (0.919) was from CNN and the lowest (0.815) was from CNN+FCN. The precision for deep neural network models ranged widely (0.156 from CNN to 0.783 from LSTM+FCN). The precision for redesigned deep neural network models was higher than that of the original models, indicating that context information such as the patient’s demographic information and drug therapeutic areas could greatly reduce the false-positive rate in predicting serendipitous drug usage. The recall ranged from 0.286 (LSTM +FCN) to 0.683 (CNN). Although CNN and CLSTM models had higher AUC and recall, their precision (0.156 and 0.172, respectively) was markedly lower than that of the other models, implying high false-positive rates in their predictions.

Our deep neural network models had higher AUCs (0.815–0.919) than SVM, random forest, and AdaBoost.M1 models that were trained with word embedding and context features (0.532–0.670) (Table V). These differences might be due to the difference in constructing word-embedding features. Deep neural networks are capable of processing unstructured features such as concatenated word-embedding vectors. However, SVM, random forest, and AdaBoost.M1 cannot handle unstructured input features, and we had to aggregate word vectors to structured machine-learning features. Drug-review comments on social media with phrases such as “it also helps” and “I used this for” were often associated with serendipitous drug usage. Therefore, losing such signals in the feature construction can impact the model performance (see section II-B).

Our deep neural network models did not perform better than SVM, random forest, and AdaBoost.M1 models that were trained with n-gram and context features. All except the CNN model had AUCs (0.815–0.866) lower than that of their counterparts (0.9–0.937) (Table V). The AUC of the CNN model (0.919) was higher than SVM with n-gram and context features, but the model had low precision (0.156), which is unfavorable for drug discovery. This finding is different from findings of several recent studies that evaluated deep neural networks and nonneural network machine-learning algorithms [19, 20]. We do not know specifically why the results differ; however, we cautiously suggest 2 possible reasons.

First, deep neural networks become increasingly complex with an increasing number of trainable parameters, e.g., weights (Table VI). Weights determine the importance of the connections among neurons in a neural network. The number of weights can be used to quantify the complexity of a deep neural network because more-complex networks have more connections between neurons. The role of weights is equivalent to the support vectors of SVM. Deep neural networks typically connect hundreds to thousands of neurons, causing the number of weights to far exceed the number of equivalent components in other machine-learning algorithms. A large number of weights requires a large volume of annotated data to train the model. Although our gold-standard dataset contained 15,714 annotated sentences, only 447 sentences were in the positive class, which might be insufficient for training CNN, LSTM, and CLSTM networks to extract patterns associated with true serendipitous drug usage. We expect that our deep neural network models will exceed their current intermediate performance if more annotated data, especially true serendipitous usage cases, become available.

Second, we used a grid search (a complete search) to find optimal hyperparameter sets for SVM, random forest, and AdaBoost.M1 algorithms. A complete search in hyperparameter space has a better chance of reaching an optimal point than a partial search approach, which we used when tuning the deep neural networks. In other words, other sets of hyperparameters may exist that would perform better with our deep neural networks than with nonneural network models with n-gram features. However, many configurations and hyperparameters can greatly affect the performance of deep neural networks by causing the search space to be too large for a complete search.

C. Potential Use of Web Applications

Serendipitous drug usage in social media can be valuable information for drug discovery and development, but they need to be manually verified to exclude false-positive cases, i.e., when patients inaccurately describe their medication outcomes. However, the massive volume of social media data makes verification challenging and time consuming. *Serendipity* can help scientists in drug discovery and development to tag, assess, filter, sort, and visualize potential serendipitous usages, so that their time and efforts can be prioritized for more promising cases.

Software developers can integrate *Serendipity* into their own applications such as patient health forums. If a user submits a drug review, the comments could be passed to our RESTful API. If the serendipitous usage prediction score exceeds a certain threshold, the patient forum could ask the user to verify whether the drug also improved a comorbid condition. In this scenario, our application could seamlessly collect and verify serendipitous drug usage from website users.

IV. Conclusion

The recent success of deep neural networks in computer vision has inspired researchers to apply these new methods to text classification [19, 20, 22, 23]. Results achieved with deep neural networks such as CNN, LSTM, and CLSTM are reported to exceed those of nonneural network algorithms such as SVM, logistic regression, and random forest [19, 20]. In the current study, we examined the ability of CNN, LSTM, and CLSTM to detect serendipitous drug usage in a gold-standard dataset that we established from a WebMD patient forum in a previous study [18]. The main challenges of this study included the use of context information from patient drug-review comments and the handling of imbalanced data (<3% of the data were positive for serendipitous drug usage).

To include context information in the analysis, we added an FCN to the original deep neural network. The 2 networks paralleled each other, with the deep neural network processing social media text and the FCN processing context information. We tried deep neural networks with n-gram features and aggregation-based word-embedding features. For the data imbalance issue, we conducted cost-sensitive learning by tuning the class weight in the model training. We also tried undersampling, oversampling, and the synthetic minority oversampling technique [60] but did not discuss them in detail because they were inferior to the other approaches described. The complex, unstructured form of natural language and the small number of serendipitous usages might be a reason, which requires future investigation.

We used our gold-standard dataset to compare original and redesigned deep neural networks with SVM, random forest, and AdaBoost.M1 algorithms. The results indicated that context information regarding a patient, disease, and medication is important for reducing false-positive rates of deep neural networks. Although deep neural networks did not perform better than the combination of the above 3 algorithms, n-grams, and context information features, they could more effectively use unstructured inputs such as concatenated word-embedding vectors. This advantage makes deep neural networks worthy of further investigation and improvement.

Our findings need to be considered in light of limitations. First, we used the default parameters of some software tools or used those explored in previous research. For example, we set the dimension of each word vector to 200 and chose convolution filters of 3 continuous sizes ($k-1$, k, and $k+1$). These settings and designs might be suboptimal for mining serendipitous drug usage from social media text. Second, we explored only CNN, LSTM, and CLSTM, but other deep learning methods are worthy of further investigation, including attention mechanism [61], transfer learning [62], and generative adversarial networks [63].

Additionally, we implemented *Serendipity*, a web-based application for people interested in mining serendipitous drug usage from social media. The application uses NLP and machine-learning models from our current and previous research [18] and currently supports 1,963 drugs. It provides a GUI to take user inputs (English-language drug reviews from various social media websites) and to visualize analytic results; it also has a RESTful API that can be integrated with other software applications. However, these efforts are just the initial phase of software development. The current implementation can be improved in several ways. First, more drugs can be added to the drug information database, possibly through integration with RxNorm [64]. Second, the GUI can provide additional options to change program behavior, such as adjusting the thresholds for sentiment and semantic difference filters or the inclusion or exclusion of certain prediction models. Finally, we recognized 2 general user groups without further validation. The next step is to more specifically identify users and generate business or real-world use cases for user studies and potential customer interviews should proceed.

References

[1]. Dudley JT, Deshpande T, and Butte AJ, “Exploiting drug–disease relationships for computational drug repositioning,” Briefings in Bioinformatics, vol. 12, pp. 303–311, 2011. [PubMed: 21690101]

[2]. Ashburn TT and Thor KB, “Drug repositioning: identifying and developing new uses for existing drugs,” Nature Review Drug Discovery, vol. 3, pp. 673–683, 2004. [PubMed: 15286734]

[3]. Yao L, Zhang Y, Li Y, Sanseau P, and Agarwal P, “Electronic health records: Implications for drug discovery,” Drug Discovery Today, vol. 16, pp. 594–599, 2011. [PubMed: 21624499]

[4]. Andronis C, Sharma A, Virvilis V, Deftereos S, and Persidis A, “Literature mining, ontologies and information visualization for drug repurposing,” Briefings in Bioinformatics, vol. 12, pp. 357–368, 2011. [PubMed: 21712342]

[5]. Keiser MJ, Setola V, Irwin JJ, Laggner C, Abbas A, Hufeisen SJ, et al., “Predicting new molecular targets for known drugs,” Nature, vol. 462, pp. 175–181, 2009. [PubMed: 19881490]

[6]. Sanseau P, Agarwal P, Barnes MR, Pastinen T, Richards JB, Cardon LR, et al., “Use of genome-wide association studies for drug repositioning,” Nature Biotechnology, vol. 30, pp. 317–320, 2012.

[7]. Gottlieb A, Stein GY, Ruppin E, and Sharan R, “PREDICT: a method for inferring novel drug indications with application to personalized medicine,” Molecular Systems Biology, vol. 7, p. 496, 2011. [PubMed: 21654673]

[8]. Wren JD, Bekeredjian R, Stewart JA, Shohet RV, and Garner HR, “Knowledge discovery by automated identification and ranking of implicit relationships,” Bioinformatics, vol. 20, pp. 389–398, 2004. [PubMed: 14960466]

[9]. Yao L, “In silico search for drug targets of natural compounds,” Current pharmaceutical biotechnology, vol. 13, pp. 1632–1639, 2012. [PubMed: 22039820]

[10]. Xu H, Aldrich MC, Chen Q, Liu H, Peterson NB, Dai Q, et al., "Validating drug repurposing signals using electronic health records: a case study of metformin associated with reduced cancer mortality," Journal of the American Medical Informatics Association, vol. 22, pp. 179–191, 2014. [PubMed: 25053577]

[11]. Ru B, Harris K, and Yao L, "A Content Analysis of Patient-Reported Medication Outcomes on Social Media", in Proceedings of IEEE 15th International Conference on Data Mining Workshops, Atlantic City, NJ, USA, 2015, pp. 472–479.

[12]. Yang CC, Jiang L, Yang H, and Tang X, "Detecting signals of adverse drug reactions from health consumer contributed content in social media", in Proceedings of ACM SIGKDD Workshop on Health Informatics (August 12, 2012), 2012.

[13]. Eshleman R and Singh R, "Leveraging graph topology and semantic context for pharmacovigilance through twitter-streams," BMC Bioinformatics, vol. 17, p. 335, 2016. [PubMed: 27766937]

[14]. Powell GE, Seifert HA, Reblin T, Burstein PJ, Blowers J, Menius JA, et al., "Social media listening for routine post-marketing safety surveillance," Drug Safety, vol. 39, pp. 443–454, 2016. [PubMed: 26798054]

[15]. Whitman CB, Reid MW, Arnold C, Patel H, Ursos L, Sa'adon R, et al., "Balancing opioid-induced gastrointestinal side effects with pain management: Insights from the online community," J Opioid Manag, vol. 11, pp. 383–91, 2015. [PubMed: 26535966]

[16]. Liu X and Chen H, "A research framework for pharmacovigilance in health social media: Identification and evaluation of patient adverse drug event reports," J Biomed Inform, vol. 58, pp. 268–79, 2015. [PubMed: 26518315]

[17]. Nikfarjam A, Sarker A, O'Connor K, Ginn R, and Gonzalez G, "Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features," J Am Med Inform Assoc, vol. 22, pp. 671–81, 2015. [PubMed: 25755127]

[18]. Ru B, Warner-Hillard C, Ge Y, and Yao L, "Identifying Serendipitous Drug Usages in Patient Forum Data", in Proceedings of the 10th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2017), Porto, Portugal, 2017, pp. 106–108.

[19]. Kim Y, "Convolutional Neural Networks for Sentence Classification", in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1746–1751.

[20]. Zhou C, Sun C, Liu Z, and Lau F, "A C-LSTM neural network for text classification," arXiv preprint arXiv:1511.08630,2015.

[21]. Li D, Liu P, Huang M, Gu Y, Zhang Y, Li X, et al., "Mapping client messages to a unified data model with mixture feature embedding convolutional neural network", in 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2017, pp. 386–391.

[22]. Li D, Huang M, Li X, Ruan Y, and Yao L, "MfeCNN: Mixture Feature Embedding Convolutional Neural Network for Data Mapping," IEEE Transactions on NanoBioscience,pp. 1–1, 2018.

[23]. Jia Y, Wang X, Cao H, Ru B, and Yang T, "An Empirical Study of Using An Ensemble Model in E-commerce Taxonomy Classification Challenge", presented at the The 2018 SIGIR Workshop On eCommerce (Accepted), Ann Arbor, MI, 2018.

[24]. Mikolov T, Sutskever I, Chen K, Corrado GS, and Dean J, "Distributed representations of words and phrases and their compositionality", in Advances in neural information processing systems, 2013, pp. 3111–3119.

[25]. Lai S, Xu L, Liu K, and Zhao J, "Recurrent Convolutional Neural Networks for Text Classification", in AAAI, 2015, pp. 2267–2273.

[26]. Graves A, "Supervised sequence labelling," in Supervised sequence labelling with recurrent neural networks, ed: Springer, 2012, pp. 5–13.

[27]. Aronson AR and Lang F-M, "An overview of MetaMap: historical perspective and recent advances," Journal of the American Medical Informatics Association, vol. 17, pp. 229–236, 2010. [PubMed: 20442139]

[28]. Manning CD, Surdeanu M, Bauer J, Finkel JR, Bethard S, and McClosky D, "The Stanford CoreNLP natural language processing toolkit", in The 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, MD, USA, 2014, pp. 55–60.

[29]. U.S. National Library of Medicine. (2016, 08/03/2015). SNOMED CT Available: http://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html

[30]. Pennington J, Socher R, and Manning C, “Glove: Global vectors for word representation”, in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[31]. Tang D, Wei F, Yang N, Zhou M, Liu T, and Qin B, “Learning sentiment-specific word embedding for twitter sentiment classification”, in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2014, pp. 1555–1565.

[32]. Rehurek R and Sojka P, “Software framework for topic modelling with large corpora”, in In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, 2010.

[33]. Chapman A (2015, 04/05/2018). Bag of Words Meets Bags of Popcorn - Use Google’s Word2Vec for movie reviews Available: https://www.kaggle.com/c/word2vec-nlp-tutorial

[34]. Cortes C, Mohri M, and Rostamizadeh A, “L 2 regularization for learning kernels”, in Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, 2009, pp. 109–116.

[35]. Mikolov T, Karafiát M, Burget L, Černocký J, and Khudanpur S, “Recurrent neural network based language model”, in Eleventh Annual Conference of the International Speech Communication Association, 2010.

[36]. Hochreiter S and Schmidhuber J, “Long short-term memory,” Neural computation, vol. 9, pp. 1735–1780, 1997. [PubMed: 9377276]

[37]. (2017, 04/05/2018). Keras: The Python Deep Learning library Available: https://keras.io

[38]. Computation using data flow graphs for scalable machine learning. Available: https://github.com/tensorflow/tensorflow

[39]. Nair V and Hinton GE, “Rectified linear units improve restricted boltzmann machines”, in Proceedings of the 27th international conference on machine learning (ICML-10), 2010, pp. 807–814.

[40]. Glorot X and Bengio Y, “Understanding the difficulty of training deep feedforward neural networks”, in Proceedings of the thirteenth international conference on artificial intelligence and statistics, 2010, pp. 249–256.

[41]. Kingma DP and Ba J, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[42]. De Boer P-T, Kroese DP, Mannor S, and Rubinstein RY, “A tutorial on the cross-entropy method,” Annals of operations research, vol. 134, pp. 19–67, 2005.

[43]. He K, Zhang X, Ren S, and Sun J, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”, in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034.

[44]. Buda M, Maki A, and Mazurowski MA, “A systematic study of the class imbalance problem in convolutional neural networks,” arXiv preprint arXiv:1710.05381, 2017.

[45]. Pumperla M (2017, 04/05/2018). Keras + Hyperopt: A very simple wrapper for convenient hyperparameter optimization Available: https://github.com/maxpumperla/hyperas

[46]. Bergstra JS, Bardenet R, Bengio Y, and Kégl B, “Algorithms for hyper-parameter optimization”, in Advances in neural information processing systems, 2011, pp. 2546–2554.

[47]. Caruana R and Niculescu-Mizil A, “Data mining in metric space: an empirical analysis of supervised learning performance criteria”, in Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, USA, 2004, pp. 69–78.

[48]. Ronacher A, Brandl G, Zapletal A, Afshar A, Edgemon C, Grind-staff C, et al., “Flask (a Python microframework),” http://flask.pocoo.org. Acessado em, vol. 6, p. 2014, 2010.

[49]. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al., “Scikit-learn: Machine learning in Python,” Journal of machine learning research, vol. 12, pp. 2825–2830, 2011.

[50]. Bostock M (2017, 12/30/2018). Data-Driven Documents Available: https://d3js.org/

[51]. Bevacqua N (2016, 12/30/2018). Tiny and blazing-fast fuzzy search in JavaScript Available: https://github.com/bevacqua/fuzzysearch

[52]. The jQuery Foundation. (12/30/2018). jQuery - wirte less, do more Available: https://jquery.com/

[53]. Krasner GE and Pope ST, “A description of the model-view-controller user interface paradigm in the smalltalk-80 system,” Journal of object oriented programming, vol. 1, pp. 26–49, 1988.

[54]. Frain B, Responsive web design with HTML5 and CSS3: Packt Publishing Ltd, 2012.

[55]. Richardson L and Ruby S, RESTful web services: “ O’Reilly Media, Inc.”, 2008.

[56]. Crockford D (2006, 12/30/2018). The JavaScript Object Notation (JSON) Available: https://www.json.org/

[57]. Owens M and Allen G, SQLite: Springer, 2010.

[58]. National Library of Medicine (US). (11/10/2017). UMLS®Reference Manual Available: https://www.ncbi.nlm.nih.gov/books/NBK9676/

[59]. Socher R, Perelygin A, Wu JY, Chuang J, Manning CD, Ng AY, et al., “Recursive deep models for semantic compositionality over a sentiment treebank”, in Proceedings of the Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 2013, pp. 1631–1642.

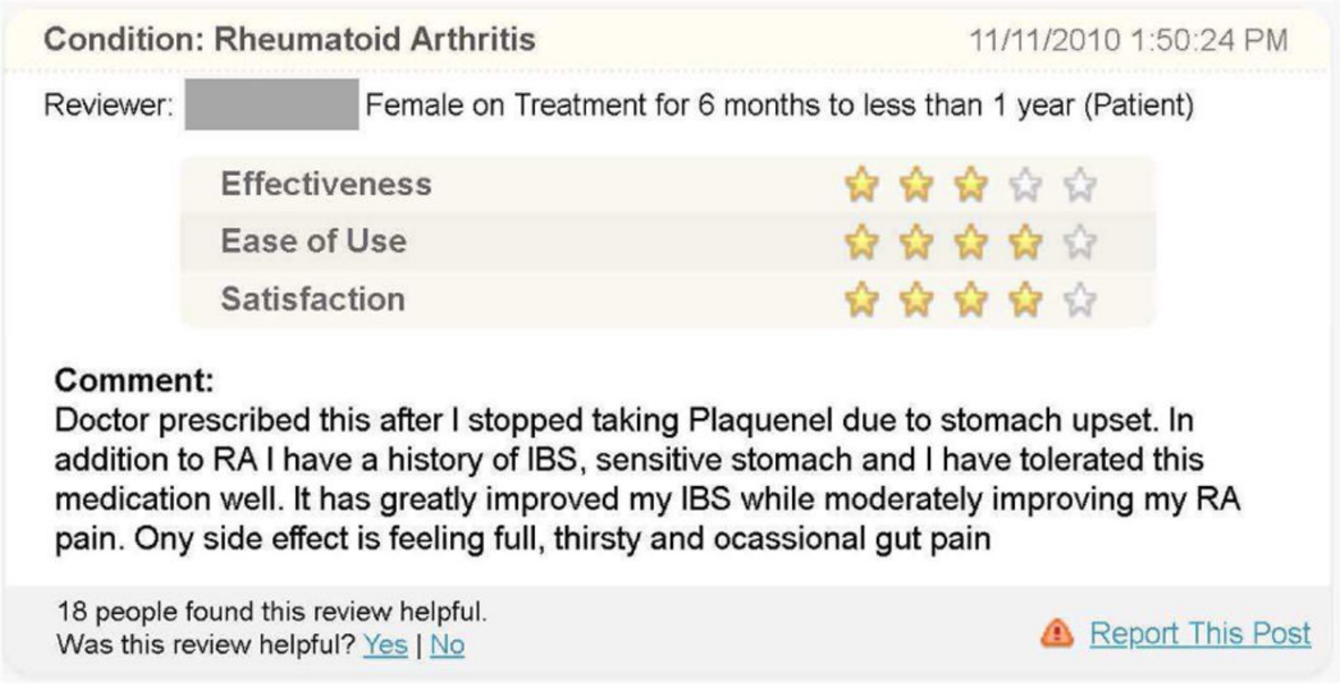
[60]. He H and Garcia EA, “Learning from imbalanced data,”IEEE Transactions on Knowledge and Data Engineering, vol. 21, pp. 1263–1284, 2009.

[61]. Yang Z, Yang D, Dyer C, He X, Smola A, and Hovy E, “Hierarchical attention networks for document classification”, in Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 1480–1489.

[62]. Do CB and Ng AY, “Transfer learning for text classification”, in Advances in Neural Information Processing Systems, 2006, pp. 299–306.

[63]. Press O, Bar A, Bogin B, Berant J, and Wolf L, “Language generation with recurrent generative adversarial networks without pre-training,” arXiv preprint arXiv:1706.01399, 2017.

[64]. Liu S, Ma W, Moore R, Ganesan V, and Nelson S, “RxNorm: prescription for electronic drug information exchange,” IT professional, vol. 7, pp. 17–23, 2005.



Condition: Rheumatoid Arthritis 11/11/2010 1:50:24 PM

Reviewer: [redacted] Female on Treatment for 6 months to less than 1 year (Patient)

Effectiveness	★★★☆☆
Ease of Use	★★★★☆
Satisfaction	★★★★☆

Comment:
Doctor prescribed this after I stopped taking Plaquenel due to stomach upset. In addition to RA I have a history of IBS, sensitive stomach and I have tolerated this medication well. It has greatly improved my IBS while moderately improving my RA pain. Ony side effect is feeling full, thirsty and ocassional gut pain

18 people found this review helpful.
Was this review helpful? Yes | No

Fig. 1.
Serendipitous drug usage in social media

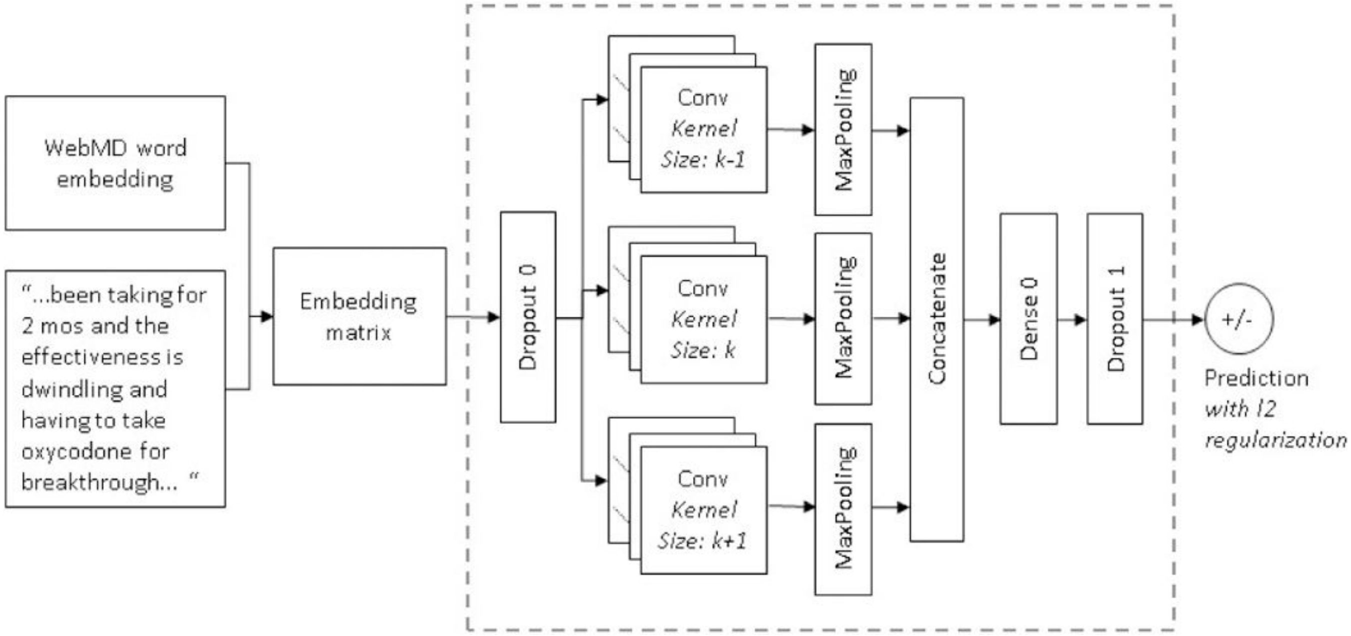


Fig. 2.

CNN model for text classification – Conv indicates convolution.

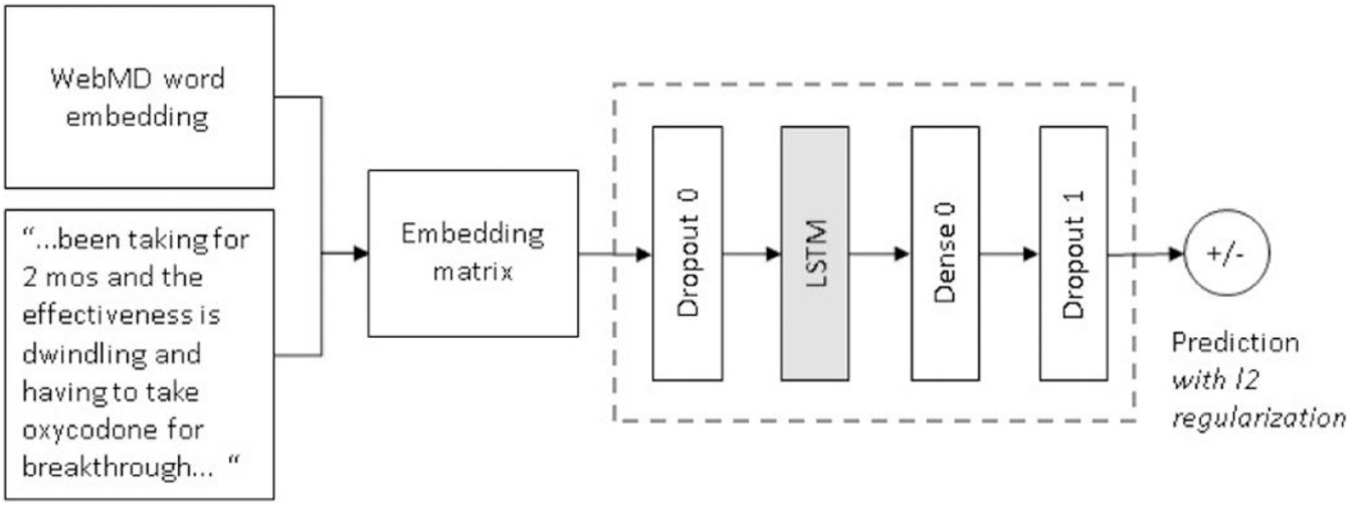

WebMD word embedding
"...been taking for 2 mos and the effectiveness is dwindling and having to take oxycodone for breakthrough... "
Embedding matrix
Dropout 0
LSTM
Dense 0
Dropout 1
+/-
Prediction with l2 regularization

Fig. 3.
CLSTM model.

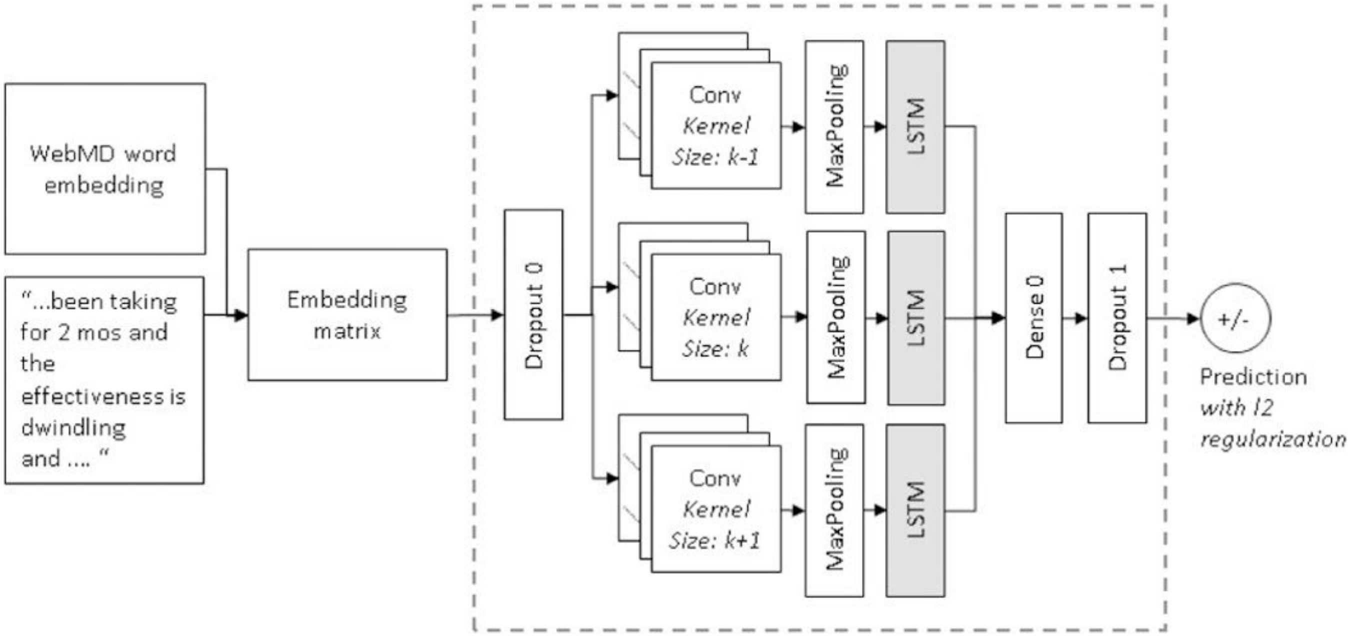


Fig. 4.
LSTM model

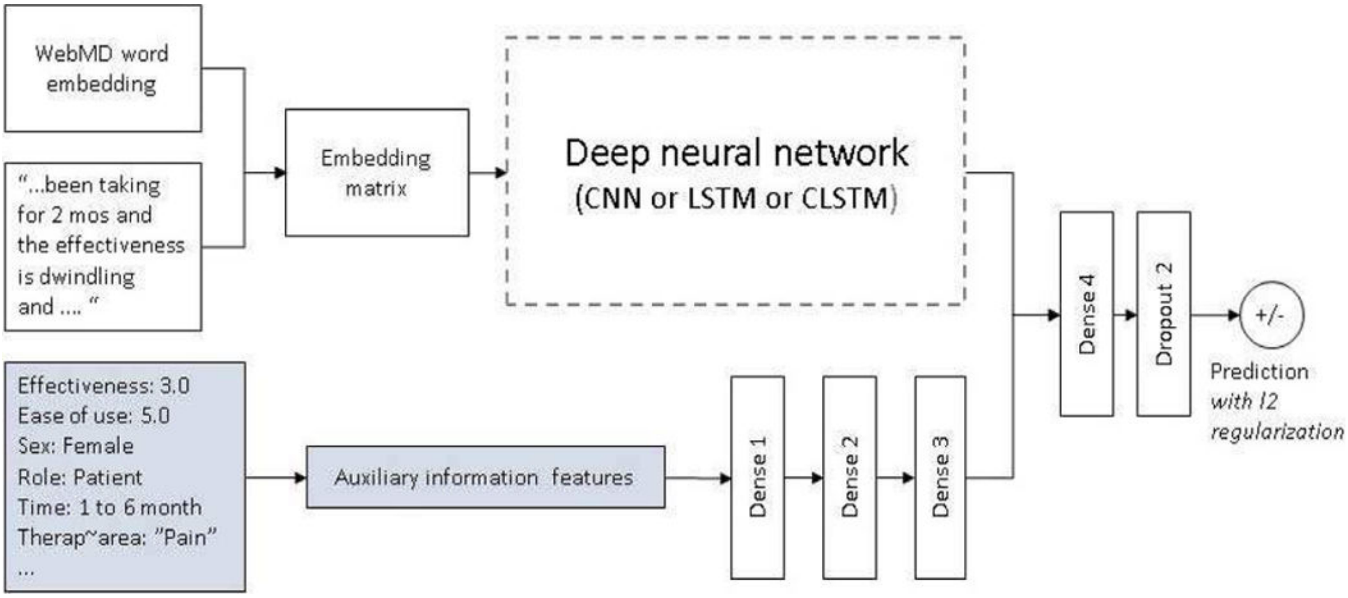


Fig. 5.
Deep neural networks with context information features

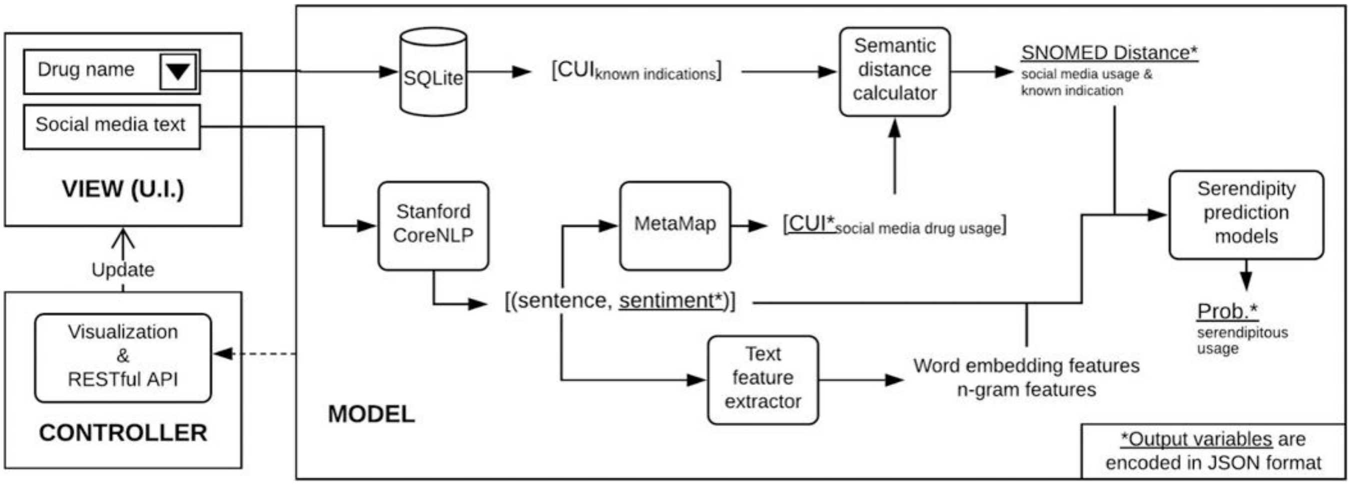
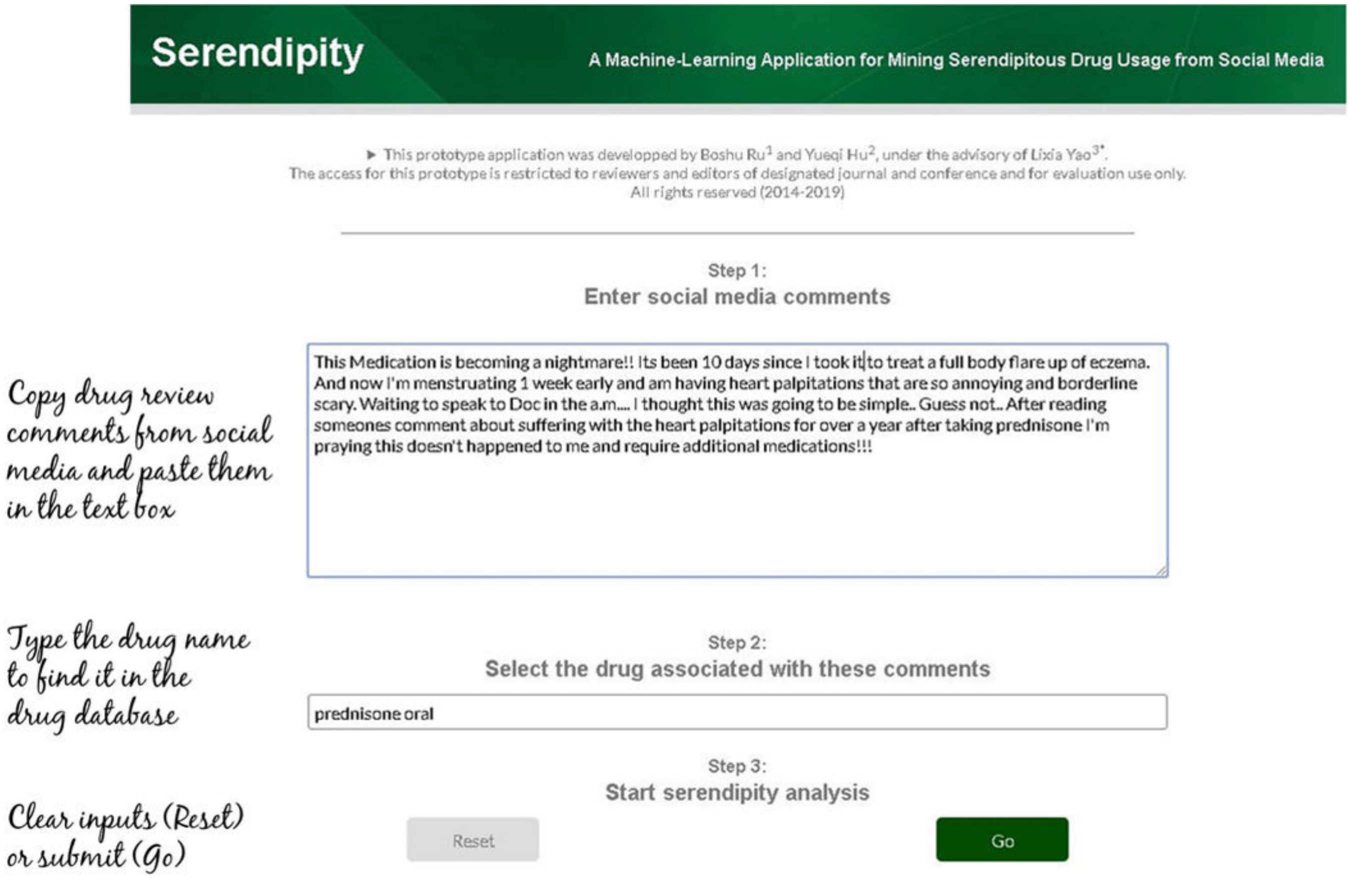


Fig. 6.

System components and workflow



Serendipity
A Machine-Learning Application for Mining Serendipitous Drug Usage from Social Media

▸ This prototype application was developped by Boshu Ru[1] and Yueqi Hu[2], under the advisory of Lixia Yao[3*].
The access for this prototype is restricted to reviewers and editors of designated journal and conference and for evaluation use only.
All rights reserved (2014-2019)

Step 1:
Enter social media comments

This Medication is becoming a nightmare!! Its been 10 days since I took it to treat a full body flare up of eczema. And now I'm menstruating 1 week early and am having heart palpitations that are so annoying and borderline scary. Waiting to speak to Doc in the a.m.... I thought this was going to be simple.. Guess not.. After reading someones comment about suffering with the heart palpitations for over a year after taking prednisone I'm praying this doesn't happened to me and require additional medications!!!

Step 2:
Select the drug associated with these comments

prednisone oral

Step 3:
Start serendipity analysis

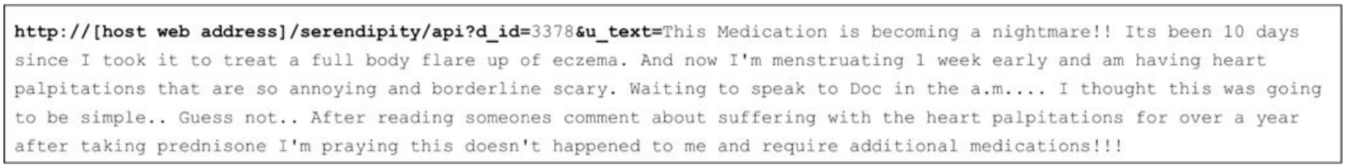
Copy drug review comments from social media and paste them in the text box

Type the drug name to find it in the drug database

Clear inputs (Reset) or submit (Go)

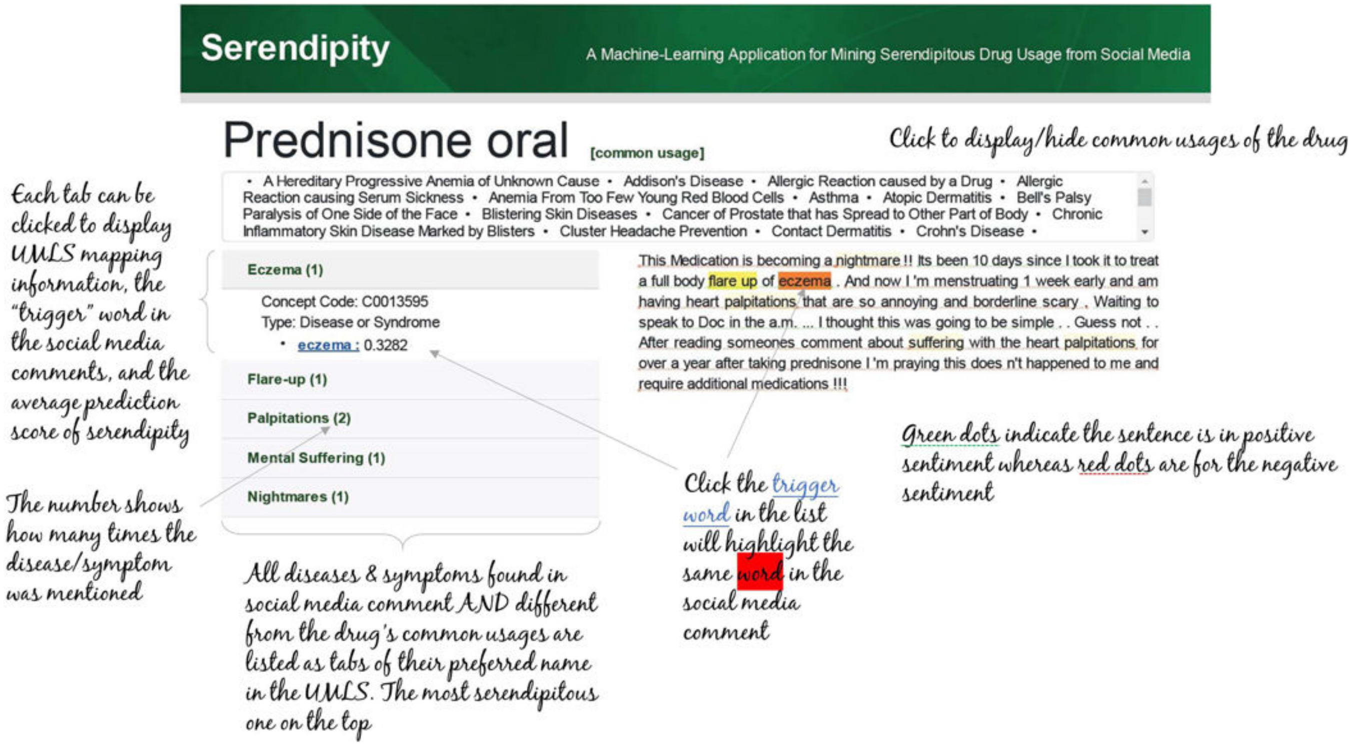
Fig. 7.

Graphic user interface



```
http://[host web address]/serendipity/api?d_id=3378&u_text=This Medication is becoming a nightmare!! Its been 10 days
since I took it to treat a full body flare up of eczema. And now I'm menstruating 1 week early and am having heart
palpitations that are so annoying and borderline scary. Waiting to speak to Doc in the a.m.... I thought this was going
to be simple.. Guess not.. After reading someones comment about suffering with the heart palpitations for over a year
after taking prednisone I'm praying this doesn't happened to me and require additional medications!!!
```

Fig. 8.
RESTful API for user input



Serendipity
A Machine-Learning Application for Mining Serendipitous Drug Usage from Social Media

Prednisone oral [common usage]

• A Hereditary Progressive Anemia of Unknown Cause • Addison's Disease • Allergic Reaction caused by a Drug • Allergic Reaction causing Serum Sickness • Anemia From Too Few Young Red Blood Cells • Asthma • Atopic Dermatitis • Bell's Palsy Paralysis of One Side of the Face • Blistering Skin Diseases • Cancer of Prostate that has Spread to Other Part of Body • Chronic Inflammatory Skin Disease Marked by Blisters • Cluster Headache Prevention • Contact Dermatitis • Crohn's Disease •

Eczema (1)

Concept Code: C0013595
Type: Disease or Syndrome
- eczema : 0.3282

Flare-up (1)

Palpitations (2)

Mental Suffering (1)

Nightmares (1)

This Medication is becoming a nightmare !! Its been 10 days since I took it to treat a full body flare up of eczema . And now I 'm menstruating 1 week early and am having heart palpitations that are so annoying and borderline scary . Waiting to speak to Doc in the a.m. ... I thought this was going to be simple . . Guess not . . After reading someones comment about suffering with the heart palpitations for over a year after taking prednisone I 'm praying this does n't happened to me and require additional medications !!!

Click to display/hide common usages of the drug

Each tab can be clicked to display UMLS mapping information, the "trigger" word in the social media comments, and the average prediction score of serendipity

The number shows how many times the disease/symptom was mentioned

All diseases & symptoms found in social media comment AND different from the drug's common usages are listed as tabs of their preferred name in the UMLS. The most serendipitous one on the top

Click the trigger word in the list will highlight the same word in the social media comment

Green dots indicate the sentence is in positive sentiment whereas red dots are for the negative sentiment

Fig. 9.

Graphic user interface for result visualization

TABLE I

Fields of the Gold-Standard Dataset

Name	Source
Drug-review sentence	WebMD
Class label	Human annotator
Auxiliary information fields	
User rating of effectiveness	WebMD
User rating of ease of use	WebMD
User rating of overall satisfaction	WebMD
Number of users who felt the review was helpful	WebMD
Number of reviews for the drug	WebMD
The day of review	WebMD
The hour of review	WebMD
User's role (e.g., patient, caregiver)	WebMD
User's sex	WebMD
User's age group	WebMD
Time on the drug	WebMD
Semantic types of medical concepts mentioned in the sentence	MetaMap
Semantic distance between the mentioned medical concepts and the drug's known indications	SNOMED
Sentiment score	Stanford CoreNLP
Therapeutic areas (155 columns of binary values)	Medical knowledge

TABLE II

Data Elements in the JSON Object

Data element	Description	Source
drug name		User
common use		SQLite database
sentence		
● text	The text of sentence	Stanford CoreNLP
● sentiment	The sentiment value encoded as numbers from −1 (very negative) to 1 (very positive)	
● concept		
○ cui	The unique identifier in UMLS	MetaMap
○ preferred term	The preferred name in UMLS	
○ trigger	The trigger word in the sentence	
○ location	The beginning index and length of characters for the trigger word in the text of sentence	
○ negation	If the trigger word is in a negation	
○ pos	The part-of-speech tag for the trigger word	
○ semantic type	The semantic type of the concept	
○ distance	The distance of concept to known drug indications in SNOMED CT, rescaled to the range between 0 (identical to any known indication of the drug) and 1 (>= 10 steps)	Semantic difference calculator
○ min distance	Flag if the concept is the most different one from known indications within the sentence (in case there are more than one concept appeared in the same sentence)	
● Prediction		
○ model_svm	The probability for the sentence to mention serendipitous drug usage	SVM model
○ model_rf		Random forest model
○ model_ada		AdaBoost.Ml model
○ model_cnn_fcn		CNN + FCN model
○ model_Istm_fcn		LSTM + FCN model
○ model_clstm_fcn		CLSTM + FCN model
○ average	The average of probabilities	

[no bullet] first-level element

• second-level element

$O_{\text{third-level element}}$

TABLE III

Model Hyperparameters

Variable		CNN	LSTM	CLSTM	CNN+FCN	LSTM+FCN	CLSTM+FCN
Convolution kernel sizes		11,12,13	--	3,4,5	7,8,9	--	5,6,7
No. of filters per kernel size		32	--	128	256	--	64
No. of units per LSTM		--	10	35	--	35	40
Size of max-pooling filter		4	--	3	5	--	2
No. of neurons	Dense_0	128	16	32	32	32	256
	Dense_l	--	--	--	256	64	256
	Dense_2[a]	--	--	--	128	32	128
	Dense_3[a]	--	--	--	64	16	64
	Dense_4	--	--	--	128	32	128
Dropout rate	Dropout_0	0.8639	0.3169	0.5657	0.53	0.1171	0.5045
	Dropout_1	0.5319	0.6546	0.3413	0.1961	0.5835	0.4131
	Dropout_2	--	--	--	0.0625	0.7011	0.1119
l2 constant		9.3467	8.6305	4.1561	3.0267	4.0071	2.2294
Initialization method		He uniform	Random uniform	Glorot uniform	He uniform	Glorot uniform	He normal
Class weights (neg : pos)		1 : 29.8666	1 : 23.7139	1 : 19.0251	1 : 25.7443	1 : 3.1155	1 : 14.1905

[a] The number of neurons for Dense_2 and Dense_3 were designed to be a half and a quarter of Dense_1, respectively.

TABLE IV

Impact of Hyperparameter

Variable		CNN	LSTM	CLSTM	CNN+FCN	LSTM+FCN	CLSTM+FCN
AUC	Minimum	0.556	0.427	0.446	0.695	0.523	0.654
	Average	0.843	0.842	0.806	0.827	0.773	0.803
	Maximum	0.908	0.909	0.908	0.908	0.892	0.894
Precision	Minimum	0.000	0.000	0.000	0.000	0.000	0.000
	Average	0.151	0.136	0.125	0.501	0.358	0.421
	Maximum	0.333	0.231	0.286	1.000	1.000	0.769
Recall	Minimum	0.000	0.000	0.000	0.000	0.000	0.000
	Average	0.239	0.499	0.389	0.331	0.299	0.363
	Maximum	0.746	0.794	1.000	0.476	0.524	0.730

TABLE V

Model Performance on the Test Dataset

Model	AUC	Precision	Recall
CNN	**0.919**	0.156	**0.683**
LSTM	0.866	0.606	0.317
CLSTM	0.871	0.172	0.635
CNN+FCN	0.815	0.735	0.397
LSTM+FCN	0.843	**0.783**	0.286
CLSTM+FCN	0.865	0.659	0.460
SVM with word embedding and context features	0.532	**0.981**	0.064
Random forest with word embedding and context features	0.651	0.905	0.302
AdaBoost.M1 with word embedding and context features	**0.670**	0.232	**0.365**
SVM with n-gram	0.900	0.758	0.397
Random forest with n-gram	0.926	**0.857**	0.381
AdaBoost.M1 with n-gram	**0.937**	0.811	**0.476**

TABLE VI

Model Complexity

Variable	CNN	LSTM	CLSTM	CNN+FCN	LSTM+FCN	CLSTM+FCN
Input dimensions	10,000	10,000	10,000	10,195	10,195	10,195
No. of CNN filters	96	--	384	768	--	192
No. of FCN filters	128	--	32	480	--	480
No. of LSTM units	--	10	135	--	35	120
No. of weights to train	345,889	8,665	380,561	1,532,833	51,297	446,561