bigNN: an open-source big data toolkit focused on biomedical sentence classification

Ahmad P. Tafti*[1], Ehsun Behravesh[2], Mehdi Assefi[3], Eric LaRose[1], Jonathan Badger[1], John Mayer[1], AnHai Doan[5], David Page[4,5], and Peggy Peissig[1]

[1]Biomedical Informatics Research Center, Marshfield Clinic Research Institute, WI 54449, USA
[2]IEEE Memebr, Kuala Lumpur, Malaysia
[3]Department of Computer Science, University of Georgia, GA 30602, USA
[4]Department of Biostatistics and Medical Informatics, University of Wisconsin-Madison, WI 53792, USA
[5]Departments of Computer Science, University of Wisconsin-Madison, WI 53706, USA

***Abstract*—Every single day, a massive amount of text data is generated by different medical data sources, such as scientific literature, medical web pages, health-related social media, clinical notes, and drug reviews. Processing this wealth of data is indeed a daunting task, and it forces us to adopt smart and scalable computational strategies, including machine intelligence, big data analytics, and distributed architecture. In this contribution, we designed and developed an open-source big data neural network toolkit, namely *bigNN* which tackles the problem of large-scale biomedical text classification in an efficient fashion, facilitating fast prototyping and reproducible text analytics researches. *bigNN* scales up a word2vec-based neural network model over Apache Spark 2.10 and Hadoop Distributed File System (HDFS) 2.7.3, allowing for more efficient big data sentence classification. The toolkit supports big data computing, and simplifies rapid application development in sentence analysis by allowing users to configure and examine different internal parameters of both Apache Spark and the neural network model. *bigNN* is fully documented, and it is publicly and freely available at https://github.com/bircatmcri/bigNN.**

***Index Terms*—Big Data Computing, Big Data Biomedical Text Classification, Open-Source Big Data Neural Network.**

I. Introduction

Every minute, hundreds of thousands of text data records are turned out through diverse medical data sources. With this extensive growth of computerized text data generated in medical literature, text categorization becomes essential to systematically manage and analyze such data sources. Text classification has been around for several years in medical and health informatics [1], [2], [3], [4], [5], and one of the most commonly used solutions for this type of problem has been supervised machine learning approach which is defined by identification of categories of new text data records, based on the probability proposed by pre-defined labeled training data. There are two major steps in text classification, feature extraction from the corpus, and classification using one of several available strategies.

The bag-of-words (BOW) known as a traditional data representation approach has been a widely used feature engineering method in text analytics, where it offers a simple text classification mechanism such that the frequency of occurrence of every single word and/or TF-IDF (term frequency-inverse document frequency) in the documents are employed as a feature vector to train a classifier (e,g., SVM, naïve Bayes, Decision Tree, Logistic Regression) [4], [6], [7], [8], [9], [5]. The BOW representation carefully keeps word intensities across all documents, but it dissembles grammar, semantics, and word order. In contrast, the word2vector (word2vec) algorithm [10] which was originally developed at Google, offers an excellent contribution to the text analytics community, because it maintains word order, grammar, and semantics within the corpus, based on a given training set. The model converts words into a well-organized vector representation model whose embedded features hold semantic meaning that boosts text classification in an accurate and reliable manner. Word2vec comes with a two-layer neural network architectural model that takes text documents as an input, and makes a set of vectors for each word in the corpus. It then groups the vectors of similar words together in a vector space, training words against other words that neighbor them in the input text documents [10], [11], [12]. The motivations of this work are to revisit the state-of-the-art in medical text classification to fulfill the following objectives: (1) Bringing an advanced neural network model to the text analytics community that assists accurate, efficient, and scalable text classification on top of the big data infrastructures, including Apache Spark and HDFS, and (2) Allowing rapid application development and fast prototyping of medical text analytics solutions by developing an open-source toolkit for the fast-growing health informatics community. We briefly summarize our main contributions as follows:

- Big data challenge our traditional computational methods in size, complexity, and velocity [13], [14], and there exists a pressing need to develop efficient tool sets to harness this wealth of data more efficiently. Inspired by the word2vec neural network model, we initiated the development of an open-source, scalable, platform independent, and highly configurable big data text classification component on top of the Apache Spark and HDFS, and made it publicly and freely available to the research community worldwide.

*Corresponding author: pahlavantafti.ahmad@marshfieldresearch.org

- Big data computing, and particularly big data neural networks are complicated in code, and their implementations are available for only a few software platforms. This practical restriction causes different difficulties to utilize them, and it makes various challenges to establish novel experiments and design new research ideas. As an important contribution, the proposed system facilitates rapid application development (RAD), and it boosts fast prototyping and reproducible research for biomedical text analytics community.
- Combining a variety of the internal neural network parameters, we presented a predictive model that obtained the auROC of 0.875 on a massive dataset downloaded from PubMed [15]. Using a moderate size of data, we were able to obtain the auROC of 0.904.
- We performed a study to compare the proposed predictive model with widely used classification algorithms, such as SVM and naïve Bayes using a BOW feature set.
- The proposed big data computing system, namely *bigNN* demonstrates the potential for mining big data scientific articles, and it offers a variety of technical advantages, such as interoperability, reusability, flexibility, and scalability, and because of it's flexibility and speed, there is an opportunity to implement near real-time classification of new articles published everyday.

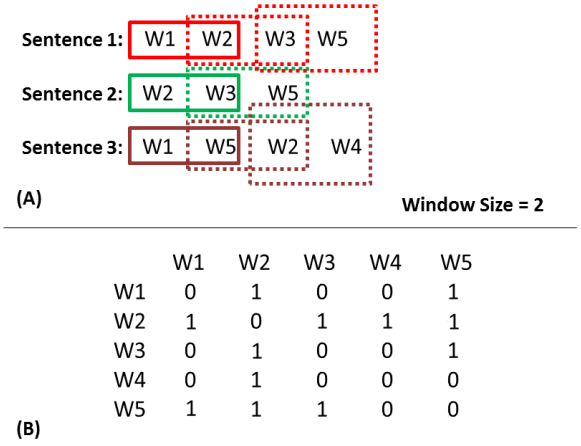
The organization of the paper is as follows. We begin by explaining the materials and methods in Section II. Experimental validations, including the test bed, dataset attributes, performance analysis, and a comparative study are described in Section III. We then further discuss the work in Section IV. Section V concludes the work and discusses future applications and research avenues.

II. MATERIALS AND METHODS

To make the paper self-contained, we shall first begin with a brief explanation of word2vec neural network model, and then delve into the software architectural model of the *bigNN*.

A. Word2vec Neural Network

Word2vec is a feedforward neural network model which has been able to tackle several text analytics problems, ranging from dependency parsing [16], [17] and named entity recognition [18], [19] to text classification [20], [21] and word clustering [22]. The model takes a text corpus as an input and creates a set of vectors for each word in the corpus. It then groups vectors of similar words together in a vector space, training words against other words that neighbor them in the input text corpus [23], [24], [25]. Word2vec is categorized into two different learning strategies: (1) Continuous bag-of-words (CBOW), and (2) Skip-gram. While the CBOW predicts a target word given a context, the skip-gram predicts a target context given a word [10]. These learning mechanisms of word2vec model are basically shallow neural networks; however, the representations acquired by them can be used in various applications of deep learning. In the skip-gram learning model, the target word is at the input layer and the context words would appear on the output layer. In the bigNN software, we used the skip-gram method since it is much more appropriate to large-scale datasets [11]. Before delving into the explanation of the word2vec skip-gram neural network model, we shall begin with the way that word2vec represents vectors of words in a corpus. Figure 1 presents an example of word2vec vector representations for five words (W1 to W5) amongst three different sentences (Sentence 1 to Sentence 3). In this example, we used a Window Size of 2, which is one of the Word2vec internal parameters that defines the context window. Using a Window Size of 2 in the example indicates that the vector of word W1 is directly affected by the word W2 and W5, and W2 can be directly affected by W1, W3, W4, and W5.



	W1	W2	W3	W4	W5
W1	0	1	0	0	1
W2	1	0	1	1	1
W3	0	1	0	0	1
W4	0	1	0	0	0
W5	1	1	1	0	0

(B)

Fig. 1. The word2vec representation vector for five distinct words across three sentences. (A) Shows three sentences including some words, and a sliding window with size of 2. (B) Shows word2vec representation vector for each word in the context. For example, the word2vec vector for W2 and W4 are 1 0 1 1 1 and 0 1 0 0 0 respectively.

Using the skip-gram method, words are read into the vector one at a time, and scanned back and forth within a certain range. Those ranges are N-grams [12], [26]. An N-gram is a contiguous sequence of N terms from a given sentence. The N-gram is then fed into a neural network to account the significance of a given word vector. A skip-gram has the training complexity architecture as follows:

$$Q = C \times (D + D \times \log_2(V)) \quad (1)$$

where C is an integer that represents the maximum distance for the words, D are word representations, and V is the dimensionality. For every training word, we will randomly choose a number R in range $< 1; C >$ and use R words from history, and R words from the future of the chosen word as the correct labels [11], [23]. This requires us to do $R \times 2$ word classifications with the selected word as input and each of the R+R words as output. By using the binary tree representation of the vocabulary, the number of output units that requires evaluation could come down to approximately $\log_2(V)$ [11] [23], [12], [26], [25], [27]. The skip-gram neural

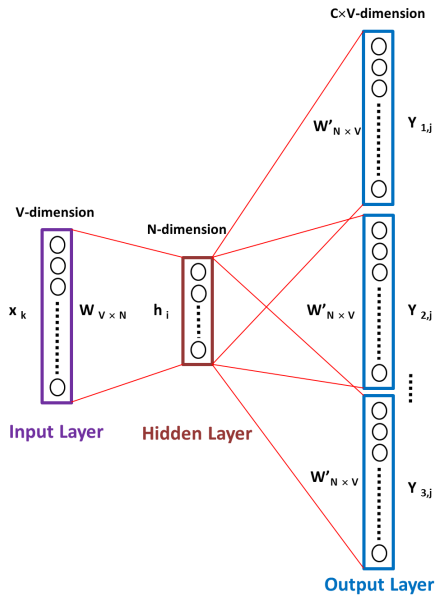


Fig. 2. The bigNN employs skip-gram architectural model to tackle the problem of large-scale sentence classification.

network architectural model is shown in Figure 2. v_{w_I} is used to define the input vector of the only word on the input layer. The weights between the input layer and the output layer could be represented by a $V \times N$ matrix W in which each row of W is the N-dimension vector representation v_w of the related word of the input layer.

Row i of W is v_w^T, and given a word (context) along with two assumptions as $x_k = 1$ and $x_{k'} = 0$ for $k' \neq k$, we will have the definition of the hidden layer outputs h as equation (2) such that it copies the k-th row of W to h. v_{w_I} is the vector representation of the input word w_I, and it shows that the activation function of the hidden layer units is linear [10], [23].

$$h = W^T x = W^T_{(k,.)} := v^T_{wI} \tag{2}$$

Here, from the hidden layer to the output layer, there will be a different weight matrix w' which is an $N \times V$ matrix as w'_{ij}. Employing the entire weights, the score u_j for every word in the context could be estimated as:

$$u_j = {v'_{w_j}}^T h \tag{3}$$

where v'_{w_j} is the j-th column of the matrix W'. In the output layer, every output is calculated using the same hidden $\rightarrow$ output matrix as equation (4).

$$p(w_{c,j} = w_{O,c}|w_I) = y_{c,j} = \frac{\exp(u_{c,j})}{\sum_{j'=1}^{V} \exp(u_{j'})} \tag{4}$$

where $w_{c,j}$ is the j-th word on the c-th panel of the output layer, $w_{O,c}$ is the exact c-th word in the output context words, w_I is the input word, $y_{c,j}$ is the output of the j-th unit on the c-th panel of the output layer, and $u_{c,j}$ is the neural network input of the j-th unit on the c-th panel of the output layer. Since the output layer panels are sharing the same weights together, therefore:

$$u_{c,j} = u_j = {v'_{w_j}}^T h, \quad for \;\; c = 1, 2, ..., C \tag{5}$$

where V'_{w_j} is the output vector of the j-th word in the vocabulary, and w_j as well as V'_{w_j} are taken from a column of the hidden $\rightarrow$ output weight matrix W'.

B. bigNN Software Architectural Model

The *bigNN* software architectural model is shown in Figure 3. One can see that the proposed architectural model deploys across disk (HDD) and main memory (RAM). The software architectural model of the proposed system consists of two different functionalities, including text pre-processing and neural network learning model. Text pre-processing involves four different tasks. Document normalization focuses on biomedical terms such as diseases (e.g., cancer types, including lung cancer, breast cancer), facts (e.g., gene, pathway), and named entities (e.g., drugs, adverse events, indications) to consistently deal with the text data towards further processing steps. Once we have a normalized text data, we need to break it up into words, terms, symbols, or elements called tokens. To avoid learning from noisy data, stop-words (e.g., all, about, across) should be eliminated, and in some cases word stemming which turns words to their word stem is required. Neural network learning components are responsible to train, test, make, and evaluate a predictive model using word2vec feedforward neural network algorithm. Text pre-processing and neural network modules run on main memory using Apache Spark 2.10, and the given labeled/unlabeled text files could be read through local file system and/or Hadoop Distributed File System (HDFS). Apache Spark [28] is an open source platform for big data processing built around speed, performance, scalability and sophisticated analytics. From the implementation perspective, the *bigNN* spreads across several packages, all implemented by Java j2SE 8 programming language. The system was designed and developed on a big data infrastructure, including Apache Hadoop cluster, Apache Spark components, and Deeplearning4j [11] which is an open-source and distributed deep learning library built for the Java community. Names and a brief description of developed packages are illustrated in Table I.

A sample output of the system is also shown in the following. As you see in the following sample output, the cosine distance similarity [29], [30] which is the normalized dot product between vectors is finally measured to find the best fitness class for a sentence.

```
Accuracy: 0.8927731092436975
Precision: 0.9230434782608696
Recall: 0.9006172839506173
====================
Document: 'ADEs' which is the file at:
home/local/ADEs_dataset/Dataset/95.txt
falls into the following categories:
ADEs Class: 0.4761768770217895
NO-ADEs Class: -0.1377755105495453
```

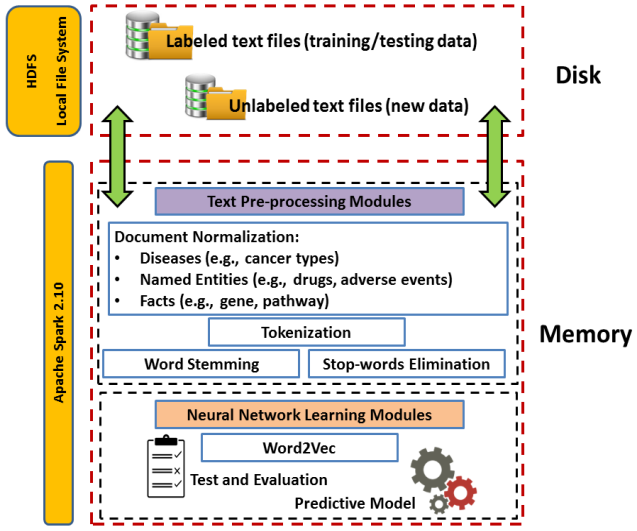


Fig. 3. The *bigNN* workflow and its architectural model. The architectural model deploys across disk and main memory. The workflow includes two major modules, the text pre-processing modules which provide text normalization, tokenization, and stemming, and the neural network learning modules that utilize the word2vec algorithm to make a predictive model. After the predictive model is trained and evaluated, it can be used on new unlabeled data records.

TABLE I
NAMES AND BRIEF DESCRIPTION OF BIGNN PACKAGES.

Package Name	Description
edu.mfldclin.mcrf.bignn.gui	Implementation of the graphical user interface
edu.mfldclin.mcrf.bignn.setting	Implementation of pre-defined and user-defined settings required to the system
edu.mfldclin.mcrf.bignn.learning	Implementation of text pre-processing and neural network learning model
edu.mfldclin.mcrf.bignn.evaluation	It evaluates the neural network predictive model

III. EXPERIMENTAL VALIDATIONS

Several extensive experiments were performed to examine the quality attributes of the *bigNN*. In this section, we utilize real data to investigate the performance of *bigNN* on sentence and also short-length text classification. What we mean by short-length text data is a summary or an abstract of a scientific article. We shall begin with introducing the test bed and the proposed datasets.

A. Test Bed

From the computational side, two VMs in a VMWARE Cluster environment, each running a 64-bit CentOS 6.8 operating system with 8 vCPUs, 16 GB RAM, and 1 TB HDD in total hosted on a Xeon E5-2690V3 2.6 GHz CPU, were used to obtain the experimental results.

B. Datasets

To examine the reliability and performance of *bigNN*, a set of text classification datasets were required. The proposed datasets are presented in Table II. The first dataset called ADEs, and it includes 21,789 sentences related to an adverse drug events (ADEs) study [31]. The dataset was divided into two different categories of sentences identified as ADEs and No-ADEs. The ADEs class refers to those sentences indicating a drug is the cause of an adverse drug event (e.g., aspirin-bleeding), and the No-ADEs class for sentences not describing a drug adverse-event relationship. The dataset was manually annotated by human experts in health informatics domain as explained in [31]. The second dataset called Human Diseases, and it includes 7,037,269 abstracts downloaded from PubMed [15], and they were related to 11 different human diseases, such as Alzheimer, Cardiovascular, Diabetes, and Cancer diseases. The abstracts were downloaded using the advanced search of PubMed which is available at [32]. The query used to download skin cancer related abstracts was as follows. Similar queries were employed for other diseases.

```
((skin cancers[MeSH Terms]) AND
("1985/01/01"[Date - MeSH] :
"2017/07/20"[Date - MeSH])
AND English[Language])
```

TABLE II
DATASETS' NAMES AND ATTRIBUTES. FOR EACH OF THE DATASET DESCRIBES HERE, WE SPLIT THE ENTIRE DATASET RANDOMLY AS 75% OF EACH CLASS TO TRAIN AND 25% TO TEST THE BIGNN SYSTEM. THE FIRST DATASET INCLUDES 21,789 INSTANCES OF ADES SENTENCES.THE SECOND DATASET INCLUDES 7,037,269 INSTANCES OF DIFFERENT HUMAN DISEASES ABSTRACTS. NUMBER OF INSTANCES IN EVERY SINGLE CLASS IS ALSO SHOWN IN THIS TABLE.

Dataset Name	Number of Classes & Instances
ADEs	**Number of Classes: 2** **Total Number of Instances: 21,789** ADEs: 10,371 No-ADEs: 11,418
Human Diseases	**Number of Classes: 11** **Total Number of Instances: 7,037,269** Asthma and Bronchial Diseases: 641,637 Cardiovascular Diseases: 2,411,012 Brain Cancer: 533,628 Breast Cancer: 251,365 Lung Cancer: 199,249 Skin Cancer: 86,621 Diabetes:718,522 Heart Diseases: 1,004,850 HIV/AIDS: 570,857 Tuberculosis: 409,211 Alzheimer: 210,317

C. Performance Analysis

We analyzed the performance of *bigNN* on the datasets illustrated in Table II. Accuracy, precision, and recall obtained by the experiment are shown in Table III. The first column shows the dataset used to accomplish every experiment. The second column describes a configuration setup of the internal *bigNN* neural network parameters. WS stands for Window Size, and it defines context windows size to generate a vector representation for words across the documents (as it was shown in Figure 1). MWF stands for minimum word frequency and it allows ignoring all words in the vocabulary with total occurrences lower than MWF value. EP stands for Epoch, and this is the number of forward and backward passes of all training examples, so that the neural network can learn from the data. ITR stands for Iteration and it

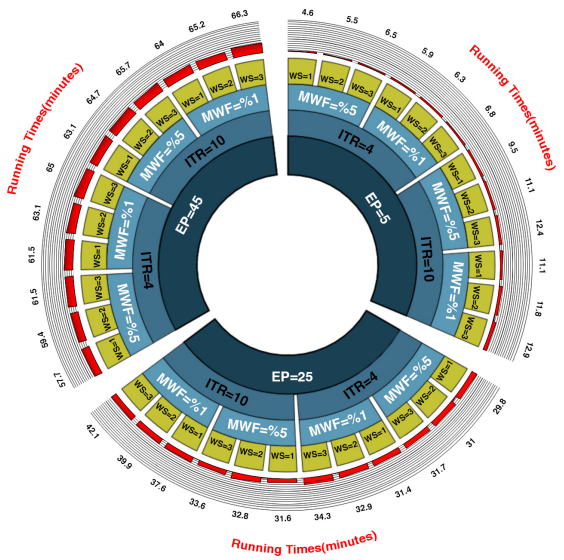


Fig. 4. The training time associated to ADEs dataset. Different *bigNN* configurations have been used to establish the current experiment. One can see the bigger EPs require more training times.

defines number of iterations done for each mini-batch during a training. This does not reflect the time of text-preprocessing tasks, such as normalization and tokenization. Figures 4 and 5 present the *bigNN* training time across ADEs and Human Diseases datasets respectively. Analyzing the training record error values against different number of training EPs is also of paramount importance to study a neural network predictive model. Figure 6 plots the training record error values against different EP values across two datasets. For the ADEs dataset, the best validation performance is 0.05383 at EP 20. (and for the Human Diseases dataset, the best validation performance is 2.812 at EP 29. The results illustrated in these experiments clearly show that a greater EP along with a greater ITR tend to be useful across both datasets, but it requires a longer training time to learn from the data. It also shows that the Windows Size (WS) depends on the problem, and it could be tuned based on the dataset size. We performed further experiments using higher value of EPs, and realized that changing the value of EPs in small quantities (e.g., 5 to 25) leads to an increase in accuracy, and the increased rates are statistically significant, while using larger quantities (e.g., 25 to 45), the impact will not be significant.

We also compared the best area under the curve (AUC) results obtained by bigNN across the datasets. Figure 7 presents this comparison. The figure shows that AUC of ADEs represents better results comparing to Human Diseases dataset. One reason could be the Human Diseases dataset suffers from a much larger vocabulary size than the ADEs.

TABLE III
PERFORMANCE ANALYSIS OF THE PROPOSED SYSTEM ACROSS BOTH DATASETS. DIFFERENT NEURAL NETWORK CONFIGURATIONS HAVE BEEN USED TO ESTABLISH THIS EXPERIMENT.

Dataset Name	bigNN Configurations	Accuracy	Precision	Recall
ADEs	WS=1; MWF=1%; EP=5; ITR=4	71.5%	70.1%	71.9%
	WS=1; MWF=1%; EP=5; ITR=10	76.3%	77.9%	75.8%
	WS=1; MWF=1%; EP=25; ITR=4	79.5%	79.2%	77.9%
	WS=1; MWF=1%; EP=25; ITR=10	82.7%	84.3%	85.1%
	WS=1; MWF=1%; EP=45; ITR=4	79.9%	80.0%	76.4%
	WS=1; MWF=1%; EP=45; ITR=10	82.9%	85.5%	84.9%
	WS=1; MWF=5%; EP=5; ITR=4	68.1%	68.5%	67.3%
	WS=1; MWF=5%; EP=5; ITR=10	73.9%	74.1%	73.5%
	WS=1; MWF=5%; EP=25; ITR=4	76.4%	75.7%	76.1%
	WS=1; MWF=5%; EP=25; ITR=10	77.3%	78.1%	78.5%
	WS=1; MWF=5%; EP=45; ITR=4	77.5%	78.0%	79.3%
	WS=1; MWF=5%; EP=45; ITR=10	78.1%	79.2%	79.0%
	WS=2; MWF=1%; EP=5; ITR=4	78.1%	81.8%	77.4%
	WS=2; MWF=1%; EP=5; ITR=10	88.3%	88.0%	88.0%
	WS=2; MWF=1%; EP=25; ITR=4	88.5%	89.7%	88.2%
	WS=2; MWF=1%; EP=25; ITR=10	91.8%	91.2%	89.3%
	WS=2; MWF=1%; EP=45; ITR=4	88.7%	88.8%	89.5%
	WS=2; MWF=1%; EP=45; ITR=10	91.5%	92.0%	89.1%
	WS=2; MWF=5%; EP=5; ITR=4	77.3%	79.7%	77.5%
	WS=2; MWF=5%; EP=5; ITR=10	87.3%	87.5%	86.7%
	WS=2; MWF=5%; EP=25; ITR=4	87.4%	88.2%	87.4%
	WS=2; MWF=5%; EP=25; ITR=10	87.5%	89.3%	87.8%
	WS=2; MWF=5%; EP=45; ITR=4	87.5%	88.0%	88.2%
	WS=2; MWF=5%; EP=45; ITR=10	88.1%	88.9%	90.3%
	WS=3; MWF=1%; EP=5; ITR=4	70.7%	68.0%	69.1%
	WS=3; MWF=1%; EP=5; ITR=10	73.2%	73.0%	74.5%
	WS=3; MWF=1%; EP=25; ITR=4	77.3%	78.9%	76.7%
	WS=3; MWF=1%; EP=25; ITR=10	79.4%	79.0%	78.5%
	WS=3; MWF=1%; EP=45; ITR=4	79.5%	80.5%	79.3%
	WS=3; MWF=1%; EP=45; ITR=10	81.2%	81.0%	80.5%
	WS=3; MWF=5%; EP=5; ITR=4	70.5%	68.4%	68.3%
	WS=3; MWF=5%; EP=5; ITR=10	72.3%	72.5%	72.6%
	WS=3; MWF=5%; EP=25; ITR=4	77.0%	77.9%	77.1%
	WS=3; MWF=5%; EP=25; ITR=10	78.5%	79.1%	78.3%
	WS=3; MWF=5%; EP=45; ITR=4	78.6%	80.2%	79.0%
	WS=3; MWF=5%; EP=45; ITR=10	80.5%	80.3%	80.0%
Human Diseases	WS=1; MWF=1%; EP=5; ITR=4	73.0%	74.1%	74.6%
	WS=1; MWF=1%; EP=5; ITR=10	73.5%	74.7%	74.8%
	WS=1; MWF=1%; EP=25; ITR=4	77.1%	78.4%	78.9%
	WS=1; MWF=1%; EP=25; ITR=10	77.5%	78.5%	78.7%
	WS=1; MWF=1%; EP=45; ITR=4	78.3%	78.5%	77.6%
	WS=1; MWF=1%; EP=45; ITR=10	78.5%	79.1%	79.5%
	WS=1; MWF=5%; EP=5; ITR=4	70.8%	72.5%	71.4%
	WS=1; MWF=5%; EP=5; ITR=10	71.6%	73.0%	72.2%
	WS=1; MWF=5%; EP=25; ITR=4	74.5%	75.0%	76.3%
	WS=1; MWF=5%; EP=25; ITR=10	74.9%	75.5%	75.9%
	WS=1; MWF=5%; EP=45; ITR=4	75.3%	75.5%	76.5%
	WS=1; MWF=5%; EP=45; ITR=10	76.0%	76.6%	76.4%
	WS=2; MWF=1%; EP=5; ITR=4	73.3%	75.0%	74.9%
	WS=2; MWF=1%; EP=5; ITR=10	73.4%	75.0%	75.4%
	WS=2; MWF=1%; EP=25; ITR=4	77.5%	78.0%	78.8%
	WS=2; MWF=1%; EP=25; ITR=10	77.7%	79.3%	78.5%
	WS=2; MWF=1%; EP=45; ITR=4	78.0%	78.1%	78.5%
	WS=2; MWF=1%; EP=45; ITR=10	78.2%	79.5%	79.8%
	WS=2; MWF=5%; EP=5; ITR=4	70.1%	70.4%	69.3%
	WS=2; MWF=5%; EP=5; ITR=10	71.5%	71.9%	72.0%
	WS=2; MWF=5%; EP=25; ITR=4	73.7%	74.6%	74.2%
	WS=2; MWF=5%; EP=25; ITR=10	73.9%	75.7%	74.8%
	WS=2; MWF=5%; EP=45; ITR=4	74.5%	75.1%	75.0%
	WS=2; MWF=5%; EP=45; ITR=10	75.7%	75.5%	74.8%
	WS=3; MWF=1%; EP=5; ITR=4	78.1%	78.7%	78.0%
	WS=3; MWF=1%; EP=5; ITR=10	79.5%	80.8%	81.0%
	WS=3; MWF=1%; EP=25; ITR=4	85.8%	87.0%	87.5%
	WS=3; MWF=1%; EP=25; ITR=10	86.5%	87.4%	88.1%
	WS=3; MWF=1%; EP=45; ITR=4	86.0%	88.1%	88.9%
	WS=3; MWF=1%; EP=45; ITR=10	86.7%	88.8%	89.2%
	WS=3; MWF=5%; EP=5; ITR=4	75.5%	76.0%	76.4%
	WS=3; MWF=5%; EP=5; ITR=10	75.9%	76.9%	76.5%
	WS=3; MWF=5%; EP=25; ITR=4	78.1%	78.5%	77.3%
	WS=3; MWF=5%; EP=25; ITR=10	78.7%	79.1%	78.4%
	WS=3; MWF=5%; EP=45; ITR=4	79.0%	79.5%	80.4%
	WS=3; MWF=5%; EP=45; ITR=10	80.6%	79.7%	81.0%

D. Comparative Study

Table IV compares the best prediction results obtained by the *bigNN* system with traditional BOW method composed with SVM, Decision Tree, naïve Bayes, and Logistic Regression classifiers. Regarding the BOW feature set, we utilized a combination of uni-grams, bi-grams, and part of speech tagging (POS) across the corpus. For each of the datasets described here, we split the data randomly as 75% to train and 25% to test the predictive model. The best accuracy results using our proposed predictive model were obtained by a con-

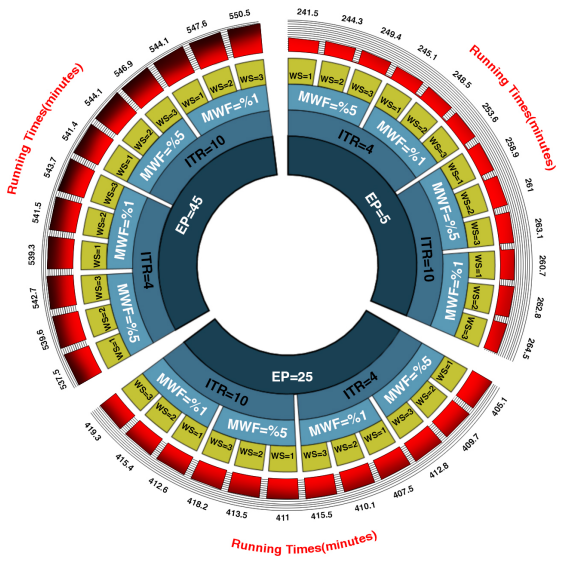


Fig. 5. The training time associated to Human Diseases dataset. Different *bigNN* configurations have been used to establish the current experiment.

TABLE IV
A COMPARATIVE STUDY OF BIGNN AND BOW MODELS. THE BEST ACCURACY RESULTS OBTAINED BY THE *bigNN* ARE COMPARED WITH TRADITIONAL BOW COMPOSED WITH SVM, DECISION TREE, NAÏVE BAYES, AND LOGISTIC REGRESSION ALGORITHMS.

Dataset Name	Learning Method	Accuracy	Precision	Recall	auROC	Training time (min.)
ADEs	**bigNN**	**91.5%**	**92.0%**	**89.1%**	**0.904**	**65.2**
	BOW+SVM	83.1%	82.7%	83.2%	0.811	84.3
	BOW+Decision Tree	80.1%	81.5%	80.4%	0.804	63.5
	BOW+Naive Bayes	81.7%	80.7%	81.9%	0.807	70.8
	BOW+logistic regression	80.1%	81.6%	81.4%	0.808	73.8
Human Diseases	**bigNN**	**86.7%**	**88.8%**	**89.2%**	**0.875**	**550.5**
	BOW+SVM	88.5%	87.1%	88.4%	0.883	611.8
	BOW+Decision Tree	81.6%	83.4%	82.0%	0.805	554.1
	BOW+Naive Bayes	82.2%	83.7%	81.6%	0.817	581.5
	BOW+logistic regression	82.4%	82.5%	82.0%	0.806	588.3

figuration as WS=2, MWF=1%, EP=45, and ITR=10 across the ADEs dataset, and with the use of WS=3, MWF=1%, EP=45, and ITR=10 on the Human Diseases dataset. Using the proposed predictive model, the vocabulary size for ADEs and Human Diseases datasets were 36,833 and 9,761,394 respectively. Using the BOW of combined uni-grams, bi-grams, and POS, the vocabulary size of ADEs and Human Diseases datasets were 75,924 and 16,733,581 respectively. For SVM, Decision Tree, naïve Bayes, and Logistic Regression classifiers, we employed Weka library (Version 3.7.12) at [33] running on hadoop-2.7 [34] by the use of HDFS (Hadoop Distributed File System). All the measures in Table IV are selected from the best performed model by tuning the models using different parameters for all the bigNN system and SVM, Decision Tree, naïve Bayes, and Logistic Regression classifiers with the BOW strategy.

We summarize the experimental validations in the following:

- A greater Epoch (EP) along with a greater Iteration (ITR) tend to be useful over the datasets, and the result using EP of 25 versus EP of 5 is statistically significant at p

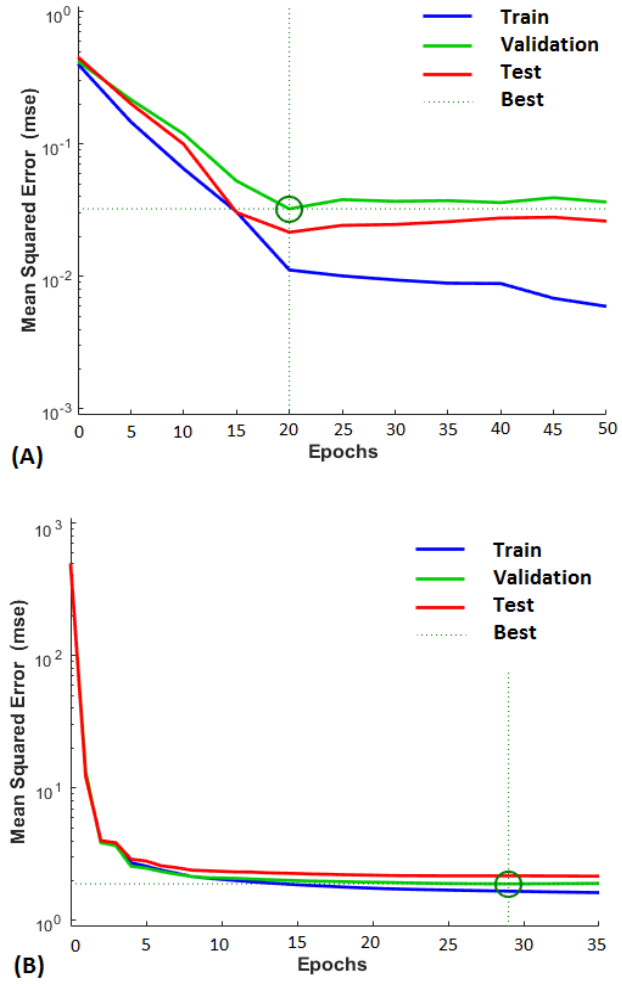


Fig. 6. The training record error values against the number of training epochs. (A) It presents the results using the ADEs dataset along with the bigNN configuration of WS=2 and MWF=1%, the best fitness parameters we've found for the ADEs dataset. The best validation performance is 0.05383 at epoch 20. (B) It shows the results employing Human Diseases dataset with the use of bigNN configuration of WS=3 and MWF=1%, the best fitness parameters we've found for the Human Disease dataset. The best validation performance is 2.812 at epoch 29.

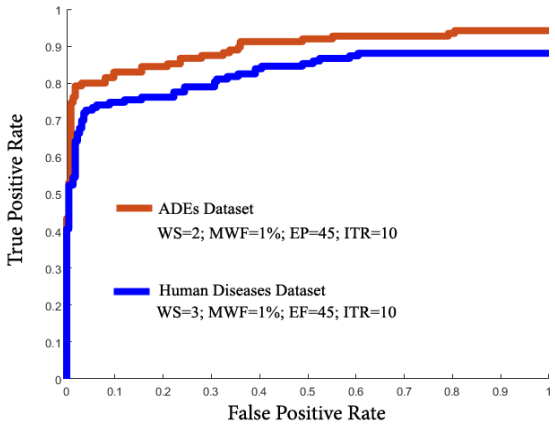


Fig. 7. The area under the curve (AUC) obtained using ADEs and Human Diseases datasets.

< 0.04 for both datasets, but it requires a longer training time.

- Increasing the value of EPs in the range of 5 to 25 epochs results in statistically significant gains in accuracy ($p < 0.05$), but gains beyond 25 epochs are much smaller and not statistically significant.
- The comparative study demonstrates that the *bigNN* is able to generate better results in comparison with traditional BOW along with SVM, Decision Tree, naïve Bayes, and/or Logistic Regression algorithms for the ADEs dataset which includes short-length text data. Performing a t-test on auROC matched by those models shows statistically significant differences (at $p < 0.05$) between our proposed model and those models developed by BOW along with traditional machine learning classifiers. It also shows no statistically differences in accuracy, precision, and recall with the use of BOW plus SVM across two datasets.
- The *bigNN* is faster than BOW method combined with SVM, Decision Tree, naïve Bayes, and Logistic Regression classification algorithms. Performing a t-test on training time matched by those methods across both datasets presents statistically significant differences (at $p < 0.05$) between the *bigNN* model and all of those four models developed by BOW composed with traditional classifiers.
- The *bigNN* performs better in classification of short-length text data (e.g., ADEs dataset which includes sentences) rather than long-length text data (e.g., Human Diseases dataset which includes abstracts of scientific articles), and the difference is statistically significant.

IV. Discussion

There exists a rapid growth in the amount of biomedical text data generated through many different mediums, and making sense of it all can be a daunting task. However, processing and understanding this information can lead to new and exciting health discoveries. Through examining self-reported findings and scientific articles, associations between medications, genomics, adverse events, and other findings can be discovered. These discoveries can benefit any number of biomedical fields. Fields such as personalized medicine, decision support, pharmacogenomics, and drug repurposing, to name a few. As these data sources continue to grow, so does the need to design and develop highly accurate and scalable text mining solutions using modern machine learning algorithms.

The main step towards a computerized and comprehensive biomedical text mining system is a high performance text classification strategy in order to automatically classify a huge number of articles into proper categories. Our proposed system, bigNN, is capable of doing just that. It is a text classification software that is designed to be both scalable and efficient. We combined several advanced computational technologies, including natural language processing (NLP), artificial neural networks, and big data infrastructure which allowed us to classify information extracted from millions of abstracts. Being built on Apache Spark components sitting over an Apache Hadoop cluster allows our software to be scalable and efficient. Combining that infrastructure with a word2vec neural network model makes *bigNN* the ideal tool for word association discovery in the realms of biomedical publications and other online mediums. When compared with traditional BOW along with different classification algorithms, *bigNN* performs better in almost every metric. This makes our contribution a competitive options for biomedical text mining.

At the heart of *bigNN* is the word2vec neural network model. This model's application has attracted attention in recent years, and many scientist and developers have already used and adapted it for research purposes [20] [21] [22] [23] [24]. The vector representations of words and phrases learned by word2vec neural network algorithm are able to keep semantic meanings and are thus very useful in many text analytics tasks. The present contribution focused on only one of many applications of word2vec neural network in text analytics, called sentence or short length text classification. In this paper, *bigNN* and its underlying algorithms and components enabled us to classify medical/health related sentences, or short length texts, namely abstracts of scientific articles, in an efficient and accurate fashion.

This work demonstrates the ability of *bigNN* to handle diverse datasets and classification problems by training the *bigNN* using both moderate and large datasets and applying binary and multi-class classification to the ADEs dataset and the Human Diseases dataset respectively. Furthermore, we were able to utilized the system to apply a binary classification of over one hundred million sentences to identify ADEs [31]. The *bigNN* is a highly customizable, robust framework that is a very important step towards knowledge discovery from biomedical text data. It demonstrates the potential for mining big data scientific articles. Because of it's flexibility and speed, there is an opportunity to implement near real-time classification of new articles published everyday.

V. Conclusion and Outlook

In this work, we design and develop *bigNN*, a tool set specializing in text classification built in a highly extensible framework that will allow for rapid text mining and classification. The proposed system offers promising results in sentence classification of short text data. It is publicly and freely available for educational, research, and academic purposes. Based on the well-organized architecture of *bigNN* and its functional attributes, it has a remarkable impact on the text analytics community by bringing the modern work2vec neural network architectural model, which is distinguished from traditional text analytics methods, while allowing big data sentence analysis.

Future work includes ways to improve the consistency of *bigNN* for sentence classification and its ability to handle long length text data classification. For classification purposes, the proposed system could incorporate TF-IDF along with other syntactic and semantic text data features plus a classifier to tackle the problem of text classification in a more reliable

way. To enhance *bigNN* in order to handle long length text data, a robust medical term tokenizer would be implemented.

Investigating additional application of *bigNN* in health informatics would also be a part of our future contributions. These contributions would lie in name entity recognition (e.g., protein and gene names), ontology extraction, information retrieval for biological processes and diseases, and pattern discovery to name a few. Through text classification, more can be learned from the novel research already being done and from patient reported findings on social media. Both of which will help to drive new initiatives and research opportunities.

With the flexibility and speed that bigNN provides, further work could be done to establish a near real-time data collection based on published articles and social media posts. In the case of ADE monitoring and discovery, this would allow researchers to identify possible drug and event correlations in an automated and efficient way. This can be essential for the development of drug safety guidelines and the protection of patient health.

VI. Acknowledgment

The project described was supported by the Clinical and Translational Science Award (CTSA) program, through the NIH National Center for Advancing Translational Sciences (NCATS), grant UL1TR000427. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH. The authors of the paper wish to thank Joseph Ellefson and Ryan Frahm at Marshfield Clinic Research Institute for their valuable contributions in providing the big data infrastructures required for this study. The authors would also like to thank Anne Nikolai at Marshfield Clinic Research Institute for her contributions to manuscript preparation.

Appendix A
Supplementary Materials

bigNN is fully documented, and it is publicly and freely available for any academic, educational, and research purposes. Supplementary materials can be found at https://github.com/bircatmcri/bigNN.

References

[1] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[2] H. Shatkay, F. Pan, A. Rzhetsky, and W. J. Wilbur, "Multi-dimensional classification of biomedical text: Toward automated, practical provision of high-utility text to diverse users," *Bioinformatics*, vol. 24, no. 18, pp. 2086–2093, 2008.

[3] A. Holzinger, J. Schantl, M. Schroettner, C. Seifert, and K. Verspoor, "Biomedical text mining: state-of-the-art, open problems and future challenges," in *Interactive Knowledge Discovery and Data Mining in Biomedical Informatics*. Springer, 2014, pp. 271–300.

[4] A. Sun, "Short text classification using very few words," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2012, pp. 1145–1146.

[5] G. H. Gonzalez, T. Tahsin, B. C. Goodale, A. C. Greene, and C. S. Greene, "Recent advances and emerging applications in text and data mining for biomedical discovery," *Briefings in bioinformatics*, vol. 17, no. 1, pp. 33–42, 2015.

[6] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1188–1196.

[7] K. Lee, D. Palsetia, R. Narayanan, M. M. A. Patwary, A. Agrawal, and A. Choudhary, "Twitter trending topic classification," in *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*. IEEE, 2011, pp. 251–258.

[8] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, "A brief survey of text mining: Classification, clustering and extraction techniques," *arXiv preprint arXiv:1707.02919*, 2017.

[9] H. Kilicoglu, "Biomedical text mining for research rigor and integrity: tasks, challenges, directions," *Briefings in Bioinformatics*, 2017.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[11] D. Team, "Deeplearning4j: Open-source distributed deep learning for the jvm," *Apache Software Foundation License*, vol. 2, 2016.

[12] E. Ordentlich, L. Yang, A. Feng, P. Cnudde, M. Grbovic, N. Djuric, V. Radosavljevic, and G. Owens, "Network-efficient distributed word2vec training system for large vocabularies," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2016, pp. 1139–1148.

[13] L. Frey, L. Lenert, and G. Lopez-Campos, "Ehr big data deep phenotyping: contribution of the imia genomic medicine working group," *Yearbook of medical informatics*, vol. 9, no. 1, p. 206, 2014.

[14] A. P. Tafti, E. LaRose, J. C. Badger, R. Kleiman, and P. Peissig, "Machine learning-as-a-service and its application to medical informatics," in *International Conference on Machine Learning and Data Mining in Pattern Recognition*. Springer, 2017, pp. 206–219.

[15] "Pubmed," https://www.ncbi.nlm.nih.gov/pubmed.

[16] M. Bansal, K. Gimpel, and K. Livescu, "Tailoring continuous word representations for dependency parsing." in *ACL (2)*, 2014, pp. 809–815.

[17] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith, "Transition-based dependency parsing with stack long short-term memory," *arXiv preprint arXiv:1505.08075*, 2015.

[18] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.

[19] C. N. d. Santos and V. Guimaraes, "Boosting named entity recognition with neural character embeddings," *arXiv preprint arXiv:1505.05008*, 2015.

[20] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and word2vec for text classification with semantic features," in *Cognitive Informatics & Cognitive Computing (ICCI* CC), 2015 IEEE 14th International Conference on*. IEEE, 2015, pp. 136–140.

[21] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[22] B. Xue, C. Fu, and Z. Shaobin, "A new clustering model based on word2vec mining on sina weibo users tags," 2014.

[23] Y. Goldberg and O. Levy, "word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722*, 2014.

[24] X. Rong, "word2vec parameter learning explained," *arXiv preprint arXiv:1411.2738*, 2014.

[25] T. Van Nguyen, A. T. Nguyen, H. D. Phan, T. D. Nguyen, and T. N. Nguyen, "Combining word2vec with revised vector space model for better code retrieval," in *Proceedings of the 39th International Conference on Software Engineering Companion*. IEEE Press, 2017, pp. 183–185.

[26] R. Ju, P. Zhou, C. H. Li, and L. Liu, "An efficient method for document categorization based on word2vec and latent semantic analysis," in *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2276–2283.

[27] S. Kottur, R. Vedantam, J. M. Moura, and D. Parikh, "Visual word2vec (vis-w2v): Learning visually grounded word embeddings using abstract scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4985–4994.

[28] "Apache spark," http://spark.apache.org/.

[29] J. Ye, "Cosine similarity measures for intuitionistic fuzzy sets and their applications," *Mathematical and Computer Modelling*, vol. 53, no. 1, pp. 91–97, 2011.

[30] L. Muflikhah and B. Baharudin, "Document clustering using concept space and cosine similarity measurement," in *Computer Technology and Development, 2009. ICCTD'09. International Conference on*, vol. 1. IEEE, 2009, pp. 58–62.
[31] A. P. Tafti, J. Badger, E. LaRose, E. Shirzadi, A. Mahnke, J. Mayer, Z. Ye, D. Page, and P. Peissig, "Adverse drug event discovery using biomedical literature: A big data neural network adventure," *JMIR medical informatics*, vol. 5, no. 4, p. e51, 2017.
[32] "Pubmed advanced search," https://www.ncbi.nlm.nih.gov/pubmed/advanced.
[33] "Weka library," http://www.cs.waikato.ac.nz/ml/weka.
[34] "Hadoop," http://hadoop.apache.org.