RT-FEND: Spark-Based Real Time FakE News Detection

Chaowei Zhang
Department of Computer Science
Yangzhou University
Yangzhou, China
007964@yzu.edu.cn

Ashish Gupta
Department of Systems and Technology
Auburn University
Auburn, USA
azg0074@auburn.edu

Hui Sun
School of Science and Technology
Anhui University
Hefei, China
sunhui@ahu.edu.cn

Yun Li
Department of Computer Science
Yangzhou University
Yangzhou, China
liyun@yzu.edu.cn

Xiao Qin
Department of Computer Science
Auburn University
Auburn, USA
xqin@auburn.edu

2022 IEEE International Conference on Networking, Architecture and Storage (NAS) | 978-1-6654-5408-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/NAS55553.2022.9925512

***Abstract*—Fake news is a rampant societal and organizational problem with various social media outlets further aggravating its spread. There is a pressing demand to assist people to identify misinformation from massive amount of news data in a timely manner. Detecting Fake news in a timely manner is critical for mitigating its impact. In this research, we propose a novel approach for detecting fake news in real time, RT-FEND (Real Time- FakE News Detection), which relies on distributed computing paradigm. The proposed methodology utilizes event and topic extraction techniques along with a topic- merging mechanism to process real time news data and reduce the number of topics for managing the curse of dimensionality. We report the findings from several experiments to compare RT-FEND with other systems to benchmark in different system settings. RT-FEND approach is more performance-improved and time-efficient in detecting fake news when compared to other fake news detection baselines.**

***Index Terms*—fake news detection, real time, text analytics, topic merging, parallel computing**

I. Introduction

Fake news fabricate information that often mimics news media contents [1]. A heated debate for 'fake news' has expanded since the presidential election of United States in 2016. Evidence from this event suggests that fake news stories were easier to be believed and more widely spread than mainstream news stories [2]. Several media outlets and reports were labeled as fake news [3]. As a consequence, authenticity has been a crucial attribute of the news with the challenge of how to distinguish and avoid fake news from news resource has been a critical problem and a hot research direction as well. Fake news stories have caused serious societal and trust issues.

One challenge in detecting fake news in a real-time manner is to deal with an excessive number of topics extracted from news articles. On average, approximately 23.1 topics are extracted from one news article. Topics tend to be diverse in different news. As a consequence, processing a large volume of news data is time consuming. To ensure the originality and integrity of news data, we stay away from the traditional dimension-reduction methods (e.g., feature pruning and principal component analysis) because topics in up-to-date news are likely to be unique.

In this study, we design a real-time fake-news detection system called *RT-FEND* running on a Spark cluster computing platform. Spark is an in-memory data processing framework built on top of the Hadoop system. Spark judiciously handles interactive queries as well as stream processing. In addition to delivering high performance, Spark offers comprehensive data analysis in the realm of machine learning and graph algorithms. Spark streaming - an extension of the core Spark APIs - enables high-throughput and fault-tolerant stream processing of live data streams. We are inspired to leverage Spark to implement a real-time parallel fake-news detection system.

Compared with an existing counterpart (i.e., *FEND*), RT-FEND is adroit at processing a massive amount of news data by embracing salient features such as topic reduction, event extraction, real-time processing, and parallel computing. We implement RT-FEND on a Spark-based parallel platform where incoming news data are collected in a real-time fashion. RT-FEND seamlessly integrates an array of processing modules to facilitate real-time data collection, news analysis, and fake-news detection. The experimental results indicate that RT-FEND achieves a high fake news detection accuracy and high efficiency.

The main contributions of this study are summarized as follows. First, we apply *WordNet* and a topic-comparison method to reduce the dimensionality of training datasets. Second, we employ the real-time analytics framework to implement streaming data collection and clustering to construct knowledge bases. Third, we develop a real-time fake-news detection model or *RT-FEND* that includes a classifying mechanism

978-1-6654-5408-7/22/$31.00 ©2022 IEEE
Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:18:57 UTC from IEEE Xplore. Restrictions apply.

based on topic and event-based filters. Fourth, we deploy RT-FEND on a up to 9 nodes computing cluster using virtual machine. Finally, we evaluate the performance of RT-FEND processing a real-world news dataset.

II. Related Works

In this section, we introduce and provide comparative description of the prevalent approaches used for fake news detection. First, we summarize the research status of fake news detection and classify into different categories based on a simple benchmark. We then analyze and discuss their relative merit and weaknesses.

A. Real-Time and Streaming Data Processing

Real-time systems are needed where information needs to be processed in a timely manner. They have been adopted in many domains such as system control [4], locating [5], deception detection [6], self-driving [7], health care analysis [8], sports [9], stock trading [10] and social media [11]. For example, use of a collection of wearable sensors referred to as Wireless Body Sensor Networks(WBSNs), patients' health states can be tracked in a real time manner. A recently developed smart phone - based WBSN, referred to as Mobile Physiological Sensor System(MoPSS), continuously collect user's physiological data from sensors embedded in a smart shirt [12]. This data is then routed via the smartphone to a remote healthcare cloud, which enables the care providers to monitor patient's conditions and make timely decisions. This is especially important for geriatric patients living in rural areas or needing assistance. Similarly, real-time detection of fake news requires several considerations, such as acquisition of news sources, timely access to news. For example, the share price in stock market changes in milliseconds and many of deception detection techniques are adopted to make predictions and detect malpractices such as spoofing. The AZFinText System predicts stock prices with a 20 minute latency using textual analysis techniques and support vector machines based on breaking Financial news and stock quotes [13].

B. Existing Fake News Detection Approaches

Fake news intentionally spreads false information that could mislead readers. Some fake news also have high credibility that causes readers to believe in them. Feature extraction has been the most popular method for fake news detection [14]. For example, [13] performed feature extraction from hoax and non-hoax posts based on a set of documents I and a set of users U using logistic regression and boolean crowdsourcing algorithms to achieve an accuracy of over 99%. Also, in [15], the author surveyed and summarized a variety of existing features based on content-specific features (i.e., author, publisher, headline, body, etc.) and social context-specific features that include user-based cues (i.e., user profiles and characteristics), post-based cues(i.e., skeptical opinions, sensational reactions, etc.), and network-based cues (i.e., followers, relationship networks, etc.).

1) Linguistic-based Methods: The goal of linguistic-based methods as applied to understand fake news is to capture the deception from articles' content. We categorize different fake news detection approaches into three groups. First Classification is based on statistical cues. This is the easiest way to obtain the feature from articles and is accomplished by counting the number of different type of elements, such as one or multi word frequency (Unigrams and Ngrams) that often involves extracting the elements from bag of words representations and using TF- IDF(Term Frequency-Inverse Document Frequency) for evaluating the importance of words or word group in the dataset [16]. Second classification is based on sentiment oriented cues. Sentiment analysis is a traditional approach and has been widely adopted for studying deception detection [17]. Intuitively, Sentiment analysis is similar to statistical analysis, both of them collect words count, but use different words categories and classification strategies for the feature extraction. Third classification is based on syntax-oriented cues. Syntax analysis has been applied to detect deception. In many cases, features based on syntax are extracted using CFG (Context-free grammar), which is used for discovering lexicalized production rules. There are other syntax analysis applications such as the Stanford CoreNLP that could be used for deception discovery and includes syntax parser relevant functions [18].

2) Network-based Methods: In many cases, network properties and behaviors may serve as critical indicators of deception. Network analytic method usually focuses on people's social network properties (e.g., credit rating, number of followers and activity records), the source of articles (i.e. fake news websites) and public knowledge base (i.e. Wikipedia Knowledge Graph). Using social network behavior cues, Cook et al. [19] verified that information propagation through twitter could be used to establish a veracity evaluation mechanism for the 2013 Australian election. Especially, the volume of retweets, hyperlinks in tweets, difference between original tweets and retweets were helpful in detecting phony online personas, fake bots and deceptive Twitter strategies. Other approaches that utilize network based fact checking depend on existing knowledge base and publicly available structured data such as DBpedia ontology [20]. Google Relation Extraction Corpus (GREC) has been widely used for deception detection [21].

In spite of effective usage of above methods, an obvious drawback of these techniques is their instability. For example, document features such as number of paragraphs, sentiment-based word count, syntax structure of sentences can be easily simulated. A valid solution is fact checking analysis which is suitable for a wide variety of textual data. A recent study has proposed a model that abandoned traditional linguistic and Network- based methods while focusing on fact extraction from datasets [22] using open information extraction techniques and refine triples as formal features i.e., events comprising of subject (u), predicate (v) and object (o) or $e = \{u, v, o\}$. Topics are represented as $t = \{u, o\}$ and, therefore, $e = \{t, v\}$. This method effectively avoids using statistical methods by

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:18:57 UTC from IEEE Xplore. Restrictions apply.

using topics as features for clustering and focuses on the meaning extraction from text, which allows the models to understand the semantic differences among articles through vectorization of features. However, their proposed approach suffers from a) efficiency (time): processing the news in a timely manner and b) extensibility (result updating): including new news items in the model. Both are a challenging research problem. Additionally, there are several more weaknesses: c) topic redundancy: different topics may have very high similarity. This study overcomes several of the drawbacks stated above by utilizing real time streaming approaches for fake news detection and benefit from a distributed computing environment.

The next section described the analytics model from an event and topics perspective. Next, we outline the framework from the perspectives of data collection and knowledge base construction. Subsequently, we describe the process of building the knowledge base. This section is followed by experiment and results.

III. The Analytic Model

In this study, a news article is modeled as a list of events extracted from completed sentences using Stanford's CoreNLP [18]. Given event lists, topics and verbs are separated from events to create features and verb lists. In what follows, we formally present the definitions of sentences, subject sets, predicate sets, object sets, and event lists.

A completed sentence includes all of the three components (Subject, Object and Predicate). A news article comprises of several sentences and, therefore, could be

$$\sigma = (U, V, O). \tag{1}$$

where U is a subject set, V is a predicate set and O is an object set.

$$U = \{u_1, u_2, ...u_p\}. \tag{2}$$

where p is the number of subjects in subject set U.

$$V = \{v_1, v_2, ...v_q\}. \tag{3}$$

where q is the number of predicates in predicate set V.

$$O = \{o_1, o_2, ...o_r\}. \tag{4}$$

where r is the number of objects in object set O.

$$x : U \times V \times O \rightarrow E. \tag{5}$$

Where E is the event set of sentence σ.

$$E = \{e_1, e_2, ...e_{pqr}\}. \tag{6}$$

The number of events in set E is equal to $p \times q \times r$.

$$e_i = \{u_i, v_i, o_i\} = \{t_i, v_i\} \tag{7}$$

where t_i is the corresponding topic of event e_i, just as equation (6), the number of topics in set E is equal to $p \times r$ (includes all of duplicated topics).

Each event e_i in set E are independent of the other events unless all the components in two contrasted events are identical. A similar assumption applies to topics. As a result, the numbers of events and topics surges dramatically along with a growing number of data points. For instance, given a data set containing a total of 14,231 news articles, the number of topics is approximately 329 thousands; the number of events exceeds 1.8 millions. To address this problem referred to as 'the curse of dimensionality', we reduce the number of topics by consolidating similar topics into a single but large topic group. This goal is achieved by applying the *WordNet* library [23] to facilitate topic similarity comparison using the following equation:

$$S_{ij} = \frac{s^u_{ij} + s^o_{ij}}{2} \tag{8}$$

where s^u_{ij} is subjects similarity between topic i and topic j, s^o_{ij} is objects similarity between topic i and topic j.

We apply a topic-merging mechanism to merge topics with high similarity in WordNet (see also Eq. 7). This mechanism is implemented by recursively (1) calculating the similarity of randomly selected 1,000 topics and (2) obtaining synonymous topics from these topics. The mechanism chooses the lowest similarity of synonymous topics as a threshold (e.g., 0.85) to prune redundant topics. This procedure of the topic-merging mechanism is better illustrated through an example. Let's consider two topics: Topic 1 includes $'investigation, solution'$ and Topic 2 includes $'survey, answer'$. *investigation* and *survey* are often used interchangeably in many textual environments; the same applies to*solution* and *answer*. More specifically, the similarity score of subjects *investigation* and *survey* is measured as 0.94 (i.e., s^u_{12}), the similarity score of objects *solution* and *answer* is quantified as 0.88 (i.e., s^o_{12}). Therefore, the similarity score between Topic 1 and Topic 2 (i.e., S_{12}) can be derived from scores s^u_{12} and s^o_{12}. Thus, the similarity between the two topics in *Example 1* is written as $S_{12} = \frac{0.94+0.88}{2} = 0.93$ (see also Eq. 8).

IV. Analytics Framework For Real Time Fake News Detection

This section describes the framework, *RT-FEND*, for real-time collecting, processing, managing and analysis of fake news. The framework (Fig. 1) contains a flow of streaming data transferred and processed in the three modules, namely, (1) real-time data collection, (2) streaming-data processing, and (3) knowledge-base construction.

News data acquisition is performed by the real-time data collection module, which forwards newly harvested data to the streaming data processing module. After processing the streaming data, the knowledge-base con- struction module builds legitimate news knowledge-base from the raw news data. The first batch of data in the Resilient Data Definitions (RDD) data structure format (see also the streaming-data

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:18:57 UTC from IEEE Xplore. Restrictions apply.

processing sub-system) makes use of an array of five data preprocessing submodules to create an initial knowledge base, which is comprised of article clusters and corresponding verb lists. The techniques implemented in the five preprocessing modules include, but not limited to, NLP-related lemmatization and stemming, events and topics extraction, and topic-based articles vectorization.

The knowledge base is repeatedly updated after subsequent data are fed into the knowledge-base construction module from preceding modules. The subsequent data are represented as the 2nd batch and the like in the streaming-data processing subsystem in Fig. 1 (see also *Time 2* to *Time n* in Fig. 1.)

A. Real Time Data Collection and Streaming-Data Processing

The data-collection module periodically acquires news data from the Internet using a web crawler, storing and archiving collected data stored in the Hadoop distributed file system (HDFS). The web crawler uses a customized algorithm to collect real-time data from a wide variety of legitimate news websites. The acquired real-time news data are dispatched to the streaming-data processing module, which converts the data format from the data-frame into the RDD. The data format conversion is critical and indispensable, because output data of the real-time data collection module is managed in the data-frame format whereas the knowledge-base construction subsystem simply handle the RDD format.

In addition to data-format conversion, another vital functionality of the streaming-data processing subsystem is to uniformly divide streaming data into equal-sized batches (see, for example, 1st and 2nd batches of data in Fig. 1). Each batch of data is scheduled and handled in a given sliding window. For example, in the streaming-data processing subsystem in Fig. 1, *time 1* and *time 2* are two sliding windows for the first and second batches of data, respectively. The first batch of data serves as an initial data point for the next subsystem (i.e., knowledge-base construction).

Given the first batch of news data in sliding window 1, an initial knowledge-base is constructed in a static manner. In other words, the RT-FEND system generates a temporary knowledge-base derived from all the news articles in the first batch of data handled in the first sliding window (a.k.a, *time 1*). Upon the arrival of the second batch of data, the knowledge-base construction module statically updates the news knowledge base by processing the second batch. Similarly, as impending batches become available thanks to the real-time data collection subsystem, all the batches of data are processed in a streaming fashion by the real-time data collection subsystem.

B. Knowledge-base Construction

We articulate the process of building a knowledge bases by transferring collected data, converting data types, and analyzing data in a high performance computing environment. As the first step, news articles that have been converted into data frames in the preceding modules are fed to this module as RDD. All RDD-formatted articles are evenly split into n

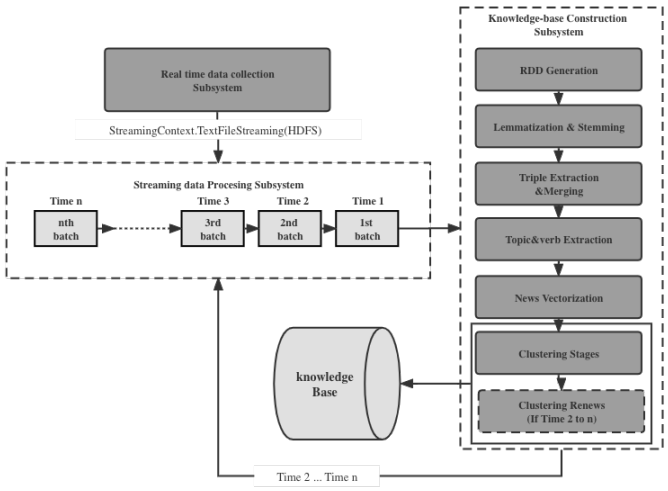


Fig. 1. A system framework for knowledge-base constructions.

pieces before dispatching to multiple slave nodes, where n is the number of slave nodes in the computing cluster. We apply *Spark* - a memory-based parallel computing framework - to fetch data into main memory for processing. We favor Spark over Hadoop thanks to Spark's high data transfer throughput in memory. Next, RDD-based raw data (i.e., news articles) are transformed as corresponding RDD-based topics and verbs (see Eqs. 5 6) and dispatch the topics and verbs into the slave nodes. Being treated as features of articles, all the topics guide the clustering process carried out in the slave nodes. The name node collects and gathers sub-results from the slave nodes, followed by grouping similar vectors into one slave for the re-clustering process. In a nutshell, the above steps help implement the data-mining algorithms for the conventional static parallel data process but for real time processing.

The streaming process mechanism is mandatory for repeatedly upgrading centroids for developing clusters. Such a mechanism allows RT-FEND to manage streaming news data to update the knowledge base in a real-time manner.

$$c_{s+1} = \frac{c_s p_s \alpha + b_s q_s}{n_s \alpha + q_s} \tag{9}$$

$$p_{s+1} = p_s + q_s \tag{10}$$

where c_s is the previous center for a news cluster, p_s is the number of points assigned to the news cluster so far, b_s is the new cluster center from the current processing batch, and q_s is the number of points added to the cluster in the current batch of data. The decay factor α controls how much past data should be incorporated in the current computing cycle. For example, if α is set to 1, than all data will be used from the beginning; if α equals to 0, than only the most recent data will be evaluated. In other words, the decay factor is analogous to an exponentially-weighted moving average [24].

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:18:57 UTC from IEEE Xplore. Restrictions apply.

V. Performance Optimization

We utilize Spark, a memory-based parallel computing framework, to fetch the data and feed into the main memory for processing. Improving the performance of memory is extremely critical for real time processing. Various memory management techniques in Spark speed up the efficiency of data processing in RT-FEND. There are two mechanisms for utilizing memory in Spark, namely, execution and storage utilization [25]. The two memory usages share unified regions of memory and acquire necessary memory according to current mission capacity. For example, when execution memory usage is low, storage memory may request all the available memory and vice versa. Execution-memory requests may evict storage memory if necessary, as long as total storage memory usage falls under a certain threshold.

A. Dataset

We compare the performance of the proposed model with another similar system, referred to as FEND [22], which is not a real time detection model but is used for fake news detection. Like FEND, RT-FEND is a two-layered fake-news detection system. The first layer (i.e., the fake-topic detection layer) detects fake news that are outliers from the perspective of the knowledge base (a.k.a., ground truth), which is comprised of data clusters built by acquiring legitimate news articles sharing similar topics. The second layer (i.e., the fake-event detection layer) is responsible for detecting fake news through an event analysis. We continue to use the same dataset as FEND for unbiased comparison. Table I summarizes the legitimate and fake news datasets.

TABLE I
Fake and legitimate news article counts from the website sources.

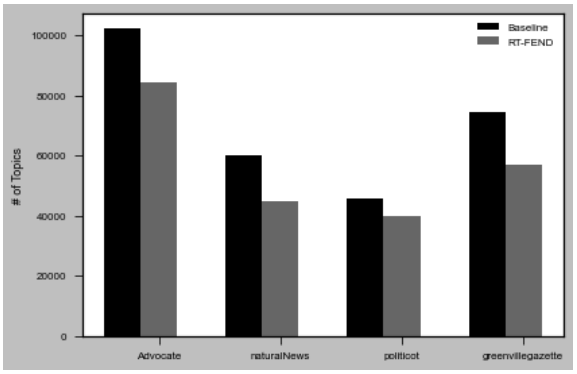
News Source	Real News	Fake News
CNN	8897	0
New York Times	5334	0
advocate [26]	0	6444
naturalnews [27]	0	2402
Politicot [28]	0	3066
greenvillegazette [29]	0	1525

VI. Experimental Results

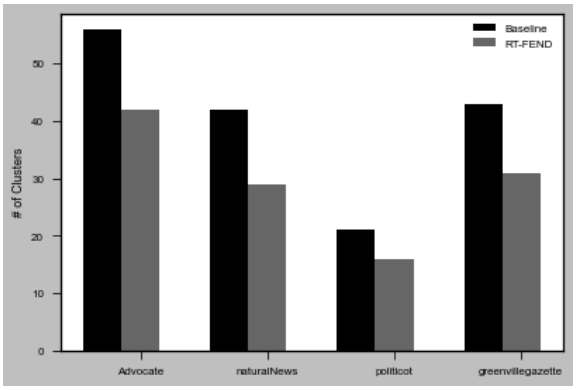
In this section, we first describe the details of an experimental testbed. Then, we carry out a group of experiments to demonstrate performance improvements from the perspectives of number of topics and number of data clusters. Next, we demonstrate the performance of RT-FEND in detecting fake news under different threshold settings. Finally, we measure RT-FEND's scalability with different system settings in terms of main memory size and the numbers of computing nodes.

A. Testbed

We apply two laptops that include 32GB 1666MHz DDR3 memory, 32GB 2133MHz DDR3 memory, an Intel 2.4Ghz quad-core *i5* processor, and an Intel 2.6Ghz quad-core *i7* processor to construct a Hadoop & Spark-based compute cluster(i.e., one master node and up to eight slave nodes) using virtual machine environment to quantitatively evaluate the performance of the proposed RT-FEND model and compare it with baseline models. The master node has the fixed configuration that includes 8GB main memory capacity and 2.6Ghz quad-core *i7* processor. The configuration of slave nodes may change from perspectives of number, CPU and memory capacities according to different experiment requirements. All the nodes are orchestrated by the Rocks system framework, which is an open-source Linux cluster distribution. Apache Spark is set up as the distributed computing framework on this cluster.



(a) the number of topics



(b) the number of clusters

Fig. 2. Comparisons of the number of topics and clusters generated by baseline FEND and RT-FEND.

B. Topic and Data Cluster Reduction

The goal of the first group of experiments is to measure the topic-reduction performance of RT-FEND. Since the baseline FEND model lacks the topic-reduction module, we evaluate the topic-reduction performance by comparing RT-FEND and FEND in terms of the number of topics as well as the number of data clusters. Evidence shows that FEND extracts approximate 200 thousand topics from 14,221 articles. These topics are independent of each other. Such an excessive number of topics inevitably triggers the high dimensionality issue in FEND. To overcome the deficiency of baseline FEND model,

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:18:57 UTC from IEEE Xplore. Restrictions apply.

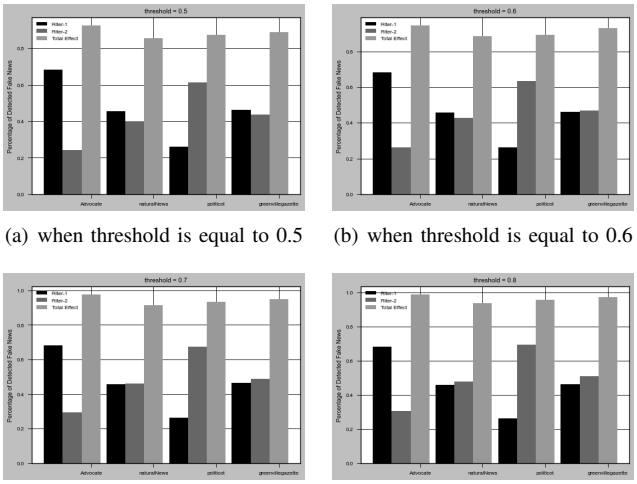
we introduced RT-FEND's topic-merging mechanism in Section III (see also Example 1 for a similarity score between two topics). In this group of experiments, the similarity threshold is set to 0.85 in the topic-merging module.

Based on the four different fake-news datasets, we illustrate the improvements of RT-FEND over FEND in terms of the number of topics (see Fig. 2(a)) and the number of data clusters (see Fig. 2(b)). Among all the four datasets, the *NaturalNews* dataset enjoys the best topic-reduction performance achieved by RT-FEND. More specifically, the topic- and cluster-reduction rates are 25.03%(i.e., 15,011 topics) and 30.95% (i.e. 13 clusters), respectively. Compared with FEND, RT-FEND reduces the number of topics in the *Advocate* dataset by 17.59% (i.e., around 18,000 reduced topics); RT-FEND slashes the number of clusters by 25% (i.e., 14 reduced clusters). For the *greenvillegazette* dataset, RT-FEND's topic- and cluster- reduction rates are 23.53% (i.e., 17,505) and 27.91% (i.e., 12). The poorest reduction performance was observed with the *politicot* dataset; nevertheless, in this case RT-FEND reduces the numbers of topics and clusters by 12.89% (i.e., 5,910) and 23.81% (i.e., 5), respectively.

In summary, RT-FEND cuts back the numbers of topics and data clusters on averages of 19.76% and 26.92%, respectively. The results confirm that RT-FEND judiciously reduces the number of topics by repeatedly (1) calculating the similarity of randomly selected topics, (2) refining synonymous topics, and (3) choosing the lowest similarity from these synonymous topics. The findings unveil that topic reductions help in cutting back the number of data clusters and; therefore, time spent in the training stage and significantly shortening the model-testing stage.

C. Performance of Fake News Detection

Fig. 3 plots the percentages of detected fake news in the two layers in the context of the four fake-news datasets; the threshold in RT-FEND is configured to 0.5, 0.6, 0.7, and 0.8, respectively. We make two intriguing observations from Fig. 3. The first observation is that regardless of threshold value, the performance of RT-FEND model largely depends on input datasets. For example, among all the four datasets, the *Advocate* dataset benefits from RT-FEND most; the *naturalNews* dataset leads to the lowest accuracy; the detection accuracies of the *politicot* and *greenvillegazette* datasets are somewhere between. The second observation is that there is no clear winner between Filter-1 and Filter-2. In the *Advocate* dataset, Filter- 1 is superior to Filter-2; in the *politicot* case, Filter-2 is better than Filter-1 in terms of accuracy; when it comes to the *naturalNews* and *greenvillegazette* datasets, Filter-1 is on par with Filter-2. These two observations suggest that the accuracy performance of RT-FEND largely depends on input datasets. There are some inherent differences among the four datasets. First, the detection rate of the filter-1 varies significantly across the four datasets. For example, the detection rate is the highest for the *advocate.com* dataset (i.e., 66.9%) and the lowest for the *politicops.com* dataset (i.e., 4.4%). Second, over 65% news in *advocate.com* are correctly detected by the first-level filter, meaning that a large portion of news posted in *advocate.com* have fake topics(detected by Filter-1). Third, the fake news detection rates by Filter-1 for *naturalnews.com* and *greenvillegazette.com* are medium low (i.e., 21.1% and 31.4%), indicating that a majority of news published on *naturalnews.com* and *greenvillegazette.com* have legitimate topics. Fourth, for the news found on *Politicot.com*, only 4.4% are detected as fake news by Filter-1, suggesting that almost of all the Politicot.com news contain trustworthy topics. Results suggest that first filtering layer provides a good mechanism for detecting the reliability of a news source. The output of the first layer is then fed to the second layer filter.



(a) when threshold is equal to 0.5 (b) when threshold is equal to 0.6

(c) when threshold is equal to 0.7 (d) when threshold is equal to 0.8

Fig. 3. Percentage of detected fake news in layer 1 (i.e., filter-1), layer 2 (i.e., filter-2), and the combined layers of filter-1 and filter 2. The threshold in RT-FEND is set to 0.5, 0.6, 0.7, and 0.8, respectively.

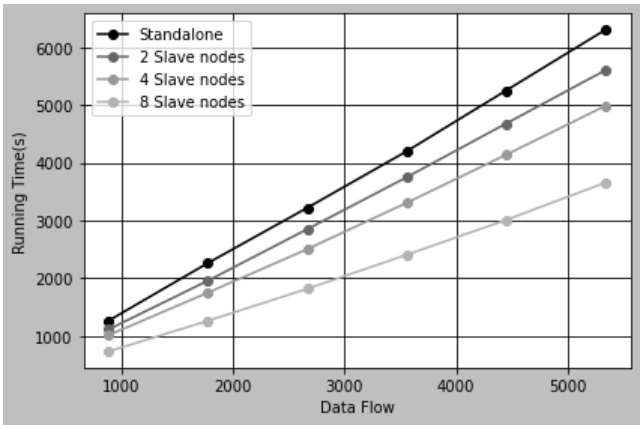
The third observation drawn from Fig. 3 is that Filter-1's performance is independent of threshold. Such an independence is expected, since the procedure used by Filter-1 for detecting fake topics is irrelevant to the threshold. Unlike Filter-1, Filter-2's performance is optimized by adjusting the threshold. In a set of news articles with legitimate topics, Filter-2 detects fake news items that comprise of fake events depending on the thresh- old value. Intuitively, Filter-2's detection performance is enhanced when the threshold value increases 3.

The fourth observation is that the second layer (i.e., filter-2) is sensitive to threshold with respect to the percentage of detected fake news; in contrast, threshold has no impact on the first layer (i.e., filter-1) in RT-FEND. Thus, an increasing number of fake news is detected by the second layer when we raise the threshold value. For example, when we push the threshold from 0.5 up to 0.6, filter-2's improvement in fake-news detection rate is 2%, 3%, 2%, and 4% for the *Advocate*, *naturalNews*, *politicot*, and *greenvillegazette* datasets, respectively. If the threshold increases from 0.6 to 0.7, such an improvement becomes 3%, 3%, 4%, and 2%, respectively. Similarly, if the threshold varies from 0.7 to 0.8,

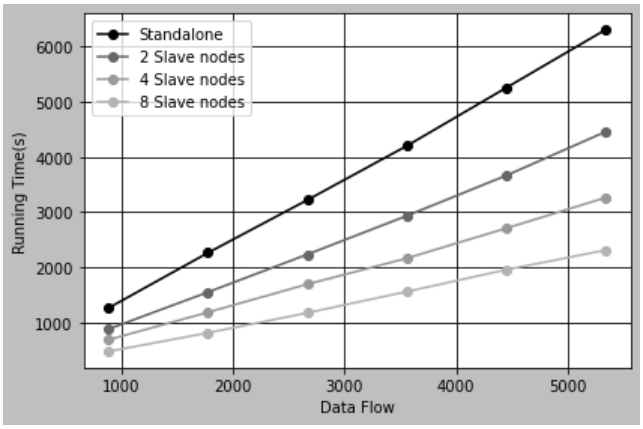
Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:18:57 UTC from IEEE Xplore. Restrictions apply.

the improvement in filter-2 will be 1%, 2%, 2%, and 2%, respectively. This performance trend implies that there is no way to optimize filter-1's detection performance through the threshold configuration; on the other end of the spectrum, one may boost filter-2's detection accuracy by adjusting the threshold.

Fig. 3 unravels RT-FEND's accuracy performance when the threshold is in the range between 0.5 and 0.8. In case of the threshold that is smaller than 0.5 or larger than 0.8, we can derive the credibility distribution of the four datasets from the results plotted in Fig. 3 . For example, in the *Advocate*, *naturalNews*, *politicot*, and *greenvillegazette* datasets, the percentage of news items with credibility lower than 0.5 is around 24%, 40%, 61%, and 44%, respectively. Given the same four datasets, the percentage of news articles with credibility larger than 0.8 becomes 1.35%, 6.49% 4.46%, and 2.92%, respectively.



(a) Slave nodes with 2GB main memory



(b) Slave nodes with 4GB main memory

Fig. 4. Impacts of the number of computing nodes and corresponding main memory sizes on the processing time of RT-FEND's preprocessing stage.

D. Scalability

This group of experiments aims to measure the scalability of RT-FEND from the perspectives of the number of computing nodes and corresponding main memory sizes based on the *New York Times* training dataset. The deployed computer cluster is formed by a master node and eight slave nodes, which means the impact of number of computing nodes is able to be evaluated by disabling some of the slave nodes. In this section, the processing time of RT-FEND model's preprocessing stage is measured to identify the impacts of number of computing nodes and corresponding main memory sizes. Specifically, we first setup the memory size of slave node with 2 Gigabytes and vary the number of slave nodes from two to eight with an increment of two nodes. In addition, the capacity of slave nodes' main memory is changed to 4 Gigabytes under the same configuration as the first experiment. These two experiments shed some lights on the impact of main memory on the model's efficiency performance.

Fig. 4 demonstrates the processing-time in preprocessing stage trends with an increasing size of training data. Each sub-figure plots four curves representing four cases, namely, a standalone system (i.e., serial computing), 2-slave nodes system, 4-slave nodes system, and 8-slave nodes system. In the second, third, and fourth cases, the number of computing nodes is set to 2, 4, and 8, respectively. In what follows, we refer to these four cases as $Standalone$, $2-slavenodes$, $4-slavenodes$, and $8-slavenodes$.

Fig. 4(a) demonstrates that the performance gaps among $Standalone$, $2-slavenodes$, $4-slavenodes$, and $8-slavenodes$ are being widened as input data size rises when the memory size of slave nodes is set to 2 Gigabytes. For example, Fig. 4(a) unravels that the $2-slavenodes$, $4-slavenodes$, and $8-slavenodes$ systems shorten the running time of the $Standalone$ system by 701, 1,318, and 2,649 seconds (i.e., from 6,306 to 5,605, 4,988, and 3,657 seconds), respectively for *New York Times* training dataset.

In contrast, Fig. 4(b) illustrates that the performance of all the four cases when the capacity of slave nodes' memory size is increased to 4 Gigabytes. Similar to Fig. 4(a), the gaps of time consuming among the four system (i.e., $Standalone$, $2-slavenodes$, $4-slavenodes$, and $8-slavenodes$) in Fig. 4(b) are 1,850, 3,043, and 3,996 seconds (i.e., from 6,306 to 4,456, 3,263, and 2,310 seconds), respectively.

Comparing Figs. 4(a) and 4(b), we observe that the performance of RT-FEND significantly improves by increasing the memory size of slave nodes. For example, in the $2-slavenodes$ case, the time consuming of the $2-slavenodes$ system decreases 1,149 seconds(i.e., from 5,605 to 4,456 seconds) while increasing the memory size from 2 Gigabytes to 4 Gigabytes. In the $4-slavenodes$ case, such processing-time reduction becomes 1,725 seconds (i.e., from 4,988 to 3,263 seconds). In the $8-slavenodes$ case, the processing time lessens from 3,657 to 2,310 seconds(i.e., 1,347 seconds cutback).

The $4-slavenodes$ case under 2 Gigabytes memory and the $2-slavenodes$ case under 4 Gigabytes memory share the same total main memory capacity. Surprisingly, the $2-slavenodes$ system outperforms the $4-slavenodes$ for such tests where input dataset varies from 0 to 5,334(the size of *New York Times* dataset). For example, comparing in the

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:18:57 UTC from IEEE Xplore. Restrictions apply.

$4-slavenodes$ case in Fig. 4(a) with the $2-slavenodes$ case in Fig. 4(b), we observe that the 2GB $4-slavenodes$ is faster than the 4GB $2-slavenodes$ by 532 seconds. Similarly, after we compare the 4GB $4-slavenodes$ case in Fig. 4(b) with $8-slavenodes$ case in Fig. 4(a), we notice that the 4GB $4-slavenodes$ system outperforms the 2GB $8-slavenodes$ system by 394 seconds (3,657 to 3,263). We conclude that keeping the overall main memory capacity as a constant, small-sized clusters for handling real time fake news are superior to large-sized counterparts due to reduced communication overhead.

We observe that when the system employs eight 2GB nodes, the system improves the performance by over 42.01% (i.e., the processing time is reduced from 6,306 to 3,657 seconds). If the system embraces eight 4GB nodes, the performance improvement becomes 63.37% (i.e., the processing time is reduced from 6,306 to 2,310 seconds) The reason for such noticeable improvements is two-fold. First, the valid usable main memory space of computing nodes is less than their entire memory space. Second, the system's overhead of assigning tasks to computing nodes is increased when the number of nodes is surging.

VII. Conclusions and Future Work

The spreading of fake news is becoming a critical social issue because of the rapid squandering of news by society. Fake news plays an intruder in form of videos, articles, or comments over social media that could damage peoples' benefits. It is urgent task to help people to promptly recognize fake news from massive news data. Most up-to-date fake news detection models lack the capability of identifying fake news in a real-time manner. To deal with the demand, we developed an NLP-based fake news detection system called *RT-FEND* that is embedded in a virtual machine-based computing cluster environment. More specifically, the proposed *RT-FEND* system is a real-time fake-news detector running on a *Spark* computing cluster. RT-FEND can provide high efficiency, scalability, and availability during the course of discovering fake news contents. In the future, RT-FEND is able to be optimized with respect to feature reduction by pruning outdated events.

References

[1] D. M. Lazer, M. A. Baum, Y. Benkler, A. J. Berinsky, K. M. Greenhill, F. Menczer, M. J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild *et al.*, "The science of fake news," *Science*, vol. 359, no. 6380, pp. 1094–1096, 2018.

[2] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of Economic Perspectives*, vol. 31, no. 2, pp. 211–36, 2017.

[3] B. McNair, *Fake news: Falsehood, fabrication and fantasy in journalism*. Routledge, 2017.

[4] S. Yin, H. Luo, and S. X. Ding, "Real-time implementation of fault-tolerant control systems with performance optimization," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 5, pp. 2402–2411, 2014.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, pp. 1137–1149, 2017.

[6] M. Tsikerdekis, "Real-time identity deception detection techniques for social media: Optimizations and challenges," *IEEE Internet Computing*, 2017.

[7] C. Kauten, A. Gupta, X. Qin, H. Li, D. Bevly, and A. Jenkins, "A perception augmentation system for autonomous vehicles," *In Pre-ICIS SIGDSA Symposium on Decision Analytics Connecting People, Data, and Things, San Francisco, CA, USA*, 2018.

[8] Y. Zhang, M. Qiu, C.-W. Tsai, M. M. Hassan, and A. Alamri, "Health-cps: Healthcare cyber-physical system assisted by cloud and big data," *IEEE Systems Journal*, vol. 11, no. 1, pp. 88–95, 2017.

[9] A. Gupta, G. B. Wilkerson, R. Sharda, and M. A. Colston, "Who is more injury-prone? prediction and assessment of injury risk," *Decision Sciences*, vol. 50, no. 2, pp. 374–409, 2019.

[10] P. D. Azar and A. W. Lo, "The wisdom of twitter crowds: Predicting stock market reactions to fomc meetings via twitter feeds," *The Journal of Portfolio Management*, vol. 42, no. 5, pp. 123–134, 2016.

[11] V. Persico, A. Pescapé, A. Picariello, and G. Sperlí, "Benchmarking big data architectures for social networks data processing using public cloud platforms," *Future Generation Computer Systems*, vol. 89, pp. 98–109, 2018.

[12] I. You, K.-K. R. Choo, C.-L. Ho *et al.*, "A smartphone-based wearable sensors for monitoring real-time physiological data," *Computers & Electrical Engineering*, vol. 65, pp. 376–392, 2018.

[13] E. Tacchini, G. Ballarin, M. L. Della Vedova, S. Moret, and L. de Alfaro, "Some like it hoax: Automated fake news detection in social networks," *arXiv preprint arXiv:1704.07506*, 2017.

[14] S. B. Parikh and P. K. Atrey, "Media-rich fake news detection: A survey," in *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 2018, pp. 436–441.

[15] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

[16] V. Rubin, N. Conroy, Y. Chen, and S. Cornwell, "Fake news or truth? using satirical cues to detect potentially misleading news," in *Proceedings of the Second Workshop on Computational Approaches to Deception Detection*, 2016, pp. 7–17.

[17] A. Gelbukh, "Sentiment analysis and opinion mining: Keynote address," in *Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO), 2017 6th International Conference on*. IEEE, 2017, pp. 41–47.

[18] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit," in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 2014, pp. 55–60.

[19] D. M. Cook, B. Waugh, M. Abdipanah, O. Hashemi, and S. A. Rahman, "Twitter deception and influence: Issues of identity, slacktivism, and puppetry," *Journal of Information Warfare*, vol. 13, no. 1, pp. 58–71, 2014.

[20] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "Dbpedia-a crystallization point for the web of data," *Web Semantics: science, services and agents on the world wide web*, vol. 7, no. 3, pp. 154–165, 2009.

[21] J. Howard and S. Ruder, "Fine-tuned language models for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[22] C. Zhang, A. Gupta, C. Kauten, A. V. Deokar, and X. Qin, "Detecting fake news for reducing misinformation risks using analytics approaches," *European Journal of Operational Research*, 2019.

[23] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[24] AMPLab, "Streaming k-means," https://github.com/amplab/iolap/blob/master/docs/mllib-clustering.md, 2016, [Online; accessed December 10, 2015].

[25] M. Zaharia, T. Das, H. Li, T. Hunter, S. Shenker, and I. Stoica, "Discretized streams: Fault-tolerant streaming computation at scale," in *Proceedings of the twenty-fourth ACM symposium on operating systems principles*. ACM, 2013, pp. 423–438.

[26] Z. Stafford, "advocate website," 1967. [Online]. Available: https://www.advocate.com/

[27] M. Adams, "naturalnews website," 2012. [Online]. Available: https://www.naturalnews.com/

[28] politycops, "politicot website," 2012. [Online]. Available: www.politicot.com

[29] Jennifer and Heather, "greenvillegazette website," 1982. [Online]. Available: http://www.greenvillegazette.com/

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:18:57 UTC from IEEE Xplore. Restrictions apply.