IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 30, 2019, accepted June 2, 2019, date of publication June 12, 2019, date of current version June 28, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2922427

Distributed Framework for Automating Opinion Discretization From Text Corpora on Facebook

HIEP XUAN HUYNH[1], VU TUAN NGUYEN[1], NGHIA DUONG-TRUNG[2,3], VAN-HUY PHAM[4], AND CANG THUONG PHAN[5]

[1]College of Information and Communications Technology, Can Tho University, Can Tho 900000, Vietnam
[2]Department of Computer Science, Can Tho University of Technology, Can Tho University, Can Tho 900000, Vietnam
[3]Department of Software Engineering, FPT University, Can Tho 900000, Vietnam
[4]NLP-KD Lab, Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh 760000, Vietnam
[5]College of Information and Communications Technology, Can Tho University, Can Tho 900000, Vietnam

Corresponding author: Van-Huy Pham (phamvanhuy@tdtu.edu.vn)

ABSTRACT Nowadays, the consecutive increase of the volume of text corpora datasets and the countless research directions in general classification have created a great opportunity and an unprecedented demand for a comprehensive evaluation of the current achievement in the research of natural language processing. There are unfortunately few studies that have applied the combination of convolutional neural networks (CNN) and Apache Spark to the task of automating opinion discretization. In this paper, the authors propose a new distributed structure for solving an opinion classification problem in text mining by utilizing CNN models and big data technologies on Vietnamese text sources. The proposed framework consists of implementation concepts that are needed by a researcher to perform experiments on text discretization problems. It covers all the steps and components that are usually part of a completely practical text mining pipeline: acquiring input data, processing, tokenizing it into a vectorial representation, applying machine learning algorithms, performing the trained models to unseen data, and evaluating their accuracy. The development of the framework started with a specific focus on binary text discretization, but soon expanded toward many other text-categorization-based problems, distributed language modeling and quantification. Several intensive assessments have been investigated to prove the robustness and efficiency of the proposed framework. Resulting in high accuracy (72.99% $\pm$ 3.64) from the experiments, one can conclude that it is feasible to perform our proposed distributed framework to the task of opinion discretization on Facebook.

INDEX TERMS Apache spark, classification, convolutional neural networks, deep learning, opinion mining, TensorFlow.

I. INTRODUCTION

Nowadays, social networks have undoubtedly become an active and vibrant ecosystem in which billions of individuals post and conduct numerous daily activities and interact with many others around the world. A social network is an Internet-based platform supporting by concept and technology of Web 2.0 that inspires individuals to create profiles, connect with other individual and publicly spread messages across different domains. Social network causes the generation and exchange of individual information. It is undoubted that social networks are an important foundation for subjectivity, opinions, online interactions, contents sharing, human behavior, sentiments expressions and many others. The prospect of cheap access and availability to a massive amount of information has encouraged great opportunities in a diverse spectrum of analytical business, academics and contexts exploitation [30]. From the research perspective, it is an interaction that embraces technologies from a wide variety of different disciplines such as data science, artificial intelligence, machine learning, optimization, mathematics, big data analytics, text mining, and the processing of natural language [47], [55]–[57].

In our research, we focus on social networks due to the exponential growth of interaction among individuals through social networks. From many existing networks, Facebook has been considered the most well-known around the globe and in Vietnam as well. Information shared on Facebook is varied from person to organizations mentioning issues such as food safety, environmental pollution, traffic accidents,

The associate editor coordinating the review of this manuscript and approving it for publication was Gang Mei.

and many others. The communication power of social networks, followed by telephone and word of mouth, has long challenged all control intentions. Social networking's information is spread through the connection between objects participating in these networks. For example on Facebook, a person can receive articles from another person if he or she is a friend or follows that person. This information continues to be shared when a user likes, shares, or tagged in the article. For each article, users can comment to express their views that are sympathetic or disagree, e.g. exposing a negative or positive opinion. The Facebook users might enunciate themselves regarding any subjects they want as the domain of Facebook posts is unlimited. Users might reveal their emotions more naturally throughout spontaneous messages. Hence, we have chosen Facebook as the subject of dataset collection because of all these reasons. The problem is how do we know the comment that the user presents is positive or negative. Therefore, building a model of explaining opinions is extremely useful for research in sentiment analysis and the process of natural language.

In the context of product reviews or opinions of different users, a considerable amount of text data has been generated on social networks. Mining such opinionated text source to reveal opinions about a subject has pervasive applications such as the recommendation of decisions and business intelligence [57]. Opinion mining is a term to describe the analysis of people's attitudes, opinion on topics, appraisals of issues, events and other individuals. The task is practically useful and technically challenging. For instance, businesses want to comprehend consumer appraisals about their services and products, e.g. whether users have a negative and/or a positive point of views. Much previous research on opinion mining has been conducted in the literature [16], [29], [33], [40]; however, these works margin their experiments on small datasets and limitation of a single computer's memory. To the best of our research, another crucial contribution of our proposed distributed framework is to conduct a discretization task in automating opinion mining over Vietnamese text sources which have not been exploited in the literature.

The organization of the remaining paper as follows. First of all, Section (II) summarizes several key research on the task of opinion mining. In Section (III), we briefly discuss the overview of technical background including big data technologies and machine learning models that summarizes a critical state-of-the-art review existing in the literature that is essential to solving the problems. Then, in Section (IV), we formalize the design concepts and introduced our proposed distributed framework to address and solve the problem in this research. In Section (V), we evaluate and perform the approach to our obtained dataset. Finally, Section (VI) recapitulates the approaches and discuss achievements done in this research.

II. STATE-OF-THE-ART RESEARCH

Supported by a sizable research community during the previous years, automatic opinion discretization has witnessed accelerate and profound improvements. Needless to say, strong industrial requirements such as rapid processing despite the exponential increment of data have guided the research in this domain. Opinion mining techniques, in general, [6], [12] have gained great attention of researchers in the field of natural language processing (or NLP in short). In a work done by [4], the authors proposed text summarization from documents by using support vector machines (SVM). Sentences from the collection of texts were labeled by the sentiment detected in the originated blog post. Then, they applied SVM classifier as their training models to classify whether a post is negative or positive. Another research direction on identifying the opinion of users toward products by exploiting positive and negative attitudes on products' characteristics has been proposed in [25]. These sentences must consist of several opinions on product attributes. Hence, this data collection obviously restricts the model's generalization. In order to classify each sentence's opinion orientation, the authors divide their proposed model into three sub-tasks. Firstly, a set of adjective words, called opinion words, is identified using an NLP method. Secondly, they determine its semantic orientation for each opinion word. Then, the WordNet [61] hierarchy is applied as a bootstrapping method. Finally, each sentence is decided its orientation of opinion. However, the use of WordNet has one crucial downside such that the words must be defined in the WordNet hierarchy. If not, these words would be ignored which leads to loss of useful information. For example, the word 'ha ha' is not determined in the WordNet but it is definitely a positive orientation. Bengali SentiWordNet dictionary is proposed to replace the original WordNet has been discussed in [2] that also supports a similar investigation. Unfortunately, in relation to language analysis, there are a few research dealing with non-English textual data. The authors in [32] have proposed a hybrid classification model which is the combination of a lexicon sentiment classifier and a machine learning algorithm. The proposed method works as follows. At the first stage, a lexicon sentiment classifier was constructed. Then, the authors used a sizable collection of labeled messages as the training set for machine learning techniques. The authors applied Naïve Bayes, SVM, and C4.5 decision trees as their investigated machine learning techniques. The approach has been developed to extract sentiment from texts written in Spanish. In a noticeable work done in [28], the authors deploy models for opinion mining that extract opinions from Twitter by turning to account big data processing technology Apache Spark and machine learning technique k-nearest neighbor. Similar to that approach, the authors in [34] have introduced recursive neural tensor network models together with semantic Treebank dictionary and their experimental results are noteworthy.

From the perspective of machine learning, a vast diversity of NLP models has been applied such as SVM, maximum entropy, and Naïve Bayes. Nevertheless, these NLP techniques do not work well in the sentiment analysis and opinion mining tasks [31]. Similar investigation done by [39] and [13]

have come up with alike conclusion. These results strongly encourage us to analyze the problem from the ground up and undertake an effective solution.

III. TECHNICAL BACKGROUND

In this Section, we introduce fundamental knowledge on several big data technologies such as TensorFlow and Apache Spark. Furthermore, the authors present the concept of deep learning and convolutional neural networks that serve as the crucial machine learning techniques applied in their work.

A. APACHE SPARK TECHNOLOGY

Apache Spark is designed as a cluster computing platform. The computation would be fast and can be generally applied in many domains [15], [38], [54]. Spark is designed to efficiently support most of the computations' types, including stream processing and queries by the extension of the popular MapReduce model [9]. Batch and streaming data can be effectively run by Apache Spark with high performance because it uses a DAG scheduler a physical execution engine. In Spark, computation is executed directly in memory which significantly increases computing speed. Consequently, batch applications, query interaction, and data streaming that required separate distributed systems [35] can be highly covered in native Spark designation. Spark also offers simple APIs, e.g. in Python, Scala, and Java, integration of other big data tools, and rich built-in libraries. Built upon the philosophy of tight integration, Spark offers the ability to seamlessly deploy applications combining different processing models. Spark is structured around Spark Core which includes components for fault recovery, memory management, optimization, task scheduling, and interacting with storage systems. Its main programming abstraction is resilient distributed datasets or called RDDs in short. Basically, an RDD is a distributed collection of elements defined by Spark Core. Spark automatically distributes the data contained in RDDs across a computing machine or a cluster of machines and parallelizes the operations performing on them. Common machine learning functionalities are defined in Spark's MLlib. Working with structured data is manipulated by using Spark SQL. Spark streaming is a Spark component that enables the processing of live streams of data. Graphs manipulation and graph-parallel computations are performed by Spark's GraphX library. Developers can deploy Spark in a stand-alone system with a Hadoop cluster supporting as the data center, or it can be deployed in association with Mesos [58]. The biggest difference between Spark and Hadoop is that Spark processes data in memory instead of a hard disk as that of Hadoop.

B. TENSORFLOW

DistBelief network project was stated in 2011 at Google as an attempt to create a general system for implementing deep neural networks. TensorFlow was released to the public in November 2015 as the second generation of DistBelief under an Apache 2.0 license. Eventually, it has been taken by the industry as a standard open source framework for deep learning [1], [45]. Going far beyond an internal Google project with its scalability and flexibility, and combining with continuous dedication and formidable commitment of Google engineers who actively dedicate their efforts to maintain and strengthen it, have made TensorFlow the leading system for developing cutting-edge real-world applications.

Tensors, basically multidimensional arrays, are the standard way of representing data in deep learning. TensorFlow refers to the flow of data and its computation along a dataflow graph. A graph refers to a set of interconnected entities, called nodes, and a set of connections called edges. An operation is done at each node, typically applies to some input, and generates an output that is passed on to other nodes via edges. Data is allowed to flow from one node to another node(s) via edge(s) in a directed manner in a dataflow graph. Nodes are dependencies in the graph. It is one of the fundamental characteristics of the graph-based computation format where developers can always identify dependencies for each node in the graph to fit their problems. Operations in a graph can include all kinds of functions, from simple addition and multiplication to more complex ones. The computations are optimized based on the graph's connectivity. In this graph, nodes represent operations, e.g. multiplication or addition, and edges indicate data, e.g. in tensor format, flowing around the system. TensorFlow has been designed with flexibility and portability, enabling these computation graphs to be executed across a wide range of operating systems and hardware architectures from a single laptop to a cluster of many high-end machines.

TensorFlow allows us to implement machine learning algorithms in a Lego-like fashion by creating and executing operations that interact with one another. These interactions assemble a computation graph which we can intuitively represent complicated functional architectures. Graph computation has featured prominently in recent research and represents one of the simplest classes of data-parallel computation that is not trivially parallelized [23]. Roughly speaking, developing TensorFlow-based models and applications involves two principal stages: (1) constructing a computation graph and (2) executing it.

C. DEEP LEARNING

From startups to large enterprises, engineers and developers are collecting huge amounts of data and apply many machine learning algorithms to address complex problems and deploy intelligent systems [3], [43]. Looking in this landscape, one can observe the category of machine learning algorithms associated with deep learning has recently gained great success across multiple disciplines. By a storm of adoption in industry and academics, deep learning is used to understand the content of images [19], [53], speech recognition [11], and many others in systems ranging from pure research [21] to real-world applications [59]. The deep architecture is inspired by the human brain's extensive network of neurons [27]. Within this architecture, we feed millions of data observations

into a network of interconnected neurons, training them to recognize patterns from these data observations. Deep neural networks are all about layers, neurons and their connections which learning a neuron is its own operation. Data enters as input and flows through the connections as it updates itself at training time or predicts outputs in a developed system. The networks take raw inputs and transform them into useful representations by adapting and correcting themselves. One of the notable advantages of deep neural networks over conventional machine learning approaches is its ability to automatically construct data representations.

One exciting research of deep learning is to build artificial intelligence systems focused on NLP and applications [5], [24], [41], [42]. Web content, social media, news, posts, and corporate correspondences have tremendously generated a huge amount of text data every day. One of the most sought-after abilities is to classify opinions into categorical classes [7], [17]. The problem of data sparsity, especially in text data, can be addressed by the idea of deep representation, and many neural networks models have been proposed for word representation. The word's neural characterization is called word embedding which we describe in more details in next sub-sections. The word embedding is a term to describe a similar measurement between words by calculating the distance of their embedding vectors.

D. CONVOLUTIONAL NEURAL NETWORKS

Convolutional neural networks, called CNNs in short, have gained extraordinary attention for a decade as an effectively promising form of deep learning [14], [36], [51], [52]. The fundamental difference between convolutional and fully connected neural networks is the pattern of connections between sequential layers. In a convolutional layer of a neural network, each node in a current layer connects to a number of nearby nodes in the previous layer. This leads to an operation known as convolution [60] which makes the name of the architecture network. Although CNNs root for classification tasks, they have found their implementation into many sub-domains of machine learning, and have been very successful for the most part. Many remarkable achievements in NLP have been conducted via utilizing convolutional neural networks models [7], [13], [20], [37]. A simple architecture of CNN for opinion discretization is presented in Figure (1).

E. BACKPROPAGATION ALGORITHM

We consider a straightforward neural network with Q layers, $q = \{1, 2, \ldots, Q\}$. While net_i is denoted as the input signal of the i^{th} neuron in q, y_i is the output signal. The neural network contains m input and n output. Furthermore, we also denote $q_{w_{ij}}$ as the parameter weight of the connection between the i neuron in q layer and the j neuron in $q-1$ layer.

In terms of artificial neural networks, an epoch refers to one cycle, both forward and backwards, through the full training dataset. An epoch is a hyperparameter which is defined before training a model. Practically, training a neural

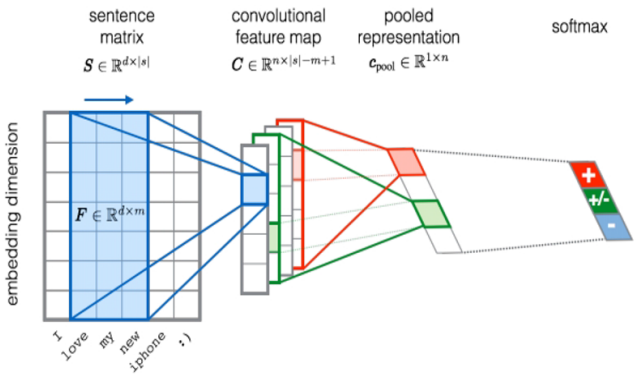

FIGURE 1. Architecture of CNN for opinion discretization.

network takes a few to thousand epochs. If we feed a neural network the training data for more than one epoch in different patterns, we hope for a better generalization when given a new "unseen" data. Given the dimensionality of data in real-world problems, it takes hundreds to thousands of epochs to get some sensible accuracy on test data. An epoch is often mixed up with an iteration. Iterations are the number of batches needed to complete one epoch. However, there is no guarantee a neural network will converge by letting it learn the data for several epochs. The convergence of the network greatly depends on the training dataset and models used. At one epoch, the calculation of the objective function is done by the following equation:

$$\text{RMS} = \sqrt{\frac{\sum_{t=1}^{p} \sum_{i=1}^{n} (y_i - d_i)^2}{t.n}} \tag{1}$$

where n and t are the numbers of parameters of the input vector and the training instances respectively. The backpropagation algorithm of our adapted interpretation is summarized in the Algorithm (1).

F. WORD EMBEDDING

With an undeniable improvement of deep learning models and techniques, it has witnessed increasing attention to training complex machine learning approaches on a massive dataset in order to solve a wide range of text mining problems. Computing distributed word representations in the form of continuous vectors [48] is a fundamental concept of such deep learning techniques. Word embedding, e.g. distributed word representations, has been widely used in various NLP problems. Word embedding seeks for representations that every token is set to a low dimensional continuous representation. Such space is assumed to convey syntactic and semantic words' information [22]. One discipline of learning word embedding representations is to discover the maximization of corpus likelihood by training neural networks.

Words are transformed into real-valued feature representations in the first layer of the network. This representation captures those words' syntactic, semantic, and morphological information. We denote an immutable-sized word

Algorithm 1 Backpropagation Algorithm

Require: training data $\{(x^k, d^k)|k = 1, 2, \ldots, p\}$, complementary vector $x^k_{m+1} = -1$

Step 0: Initializing $\mu > 0, E_{\max}, E = 0, k = 1$

Step 1: Updating parameters
Randomly select training observation k
At the input layer: $q_{y_i} = 1_{y_i} = x^k_i, \forall i$

Step 2: Propagating signals to the last layer
$q_{y_i} = g(q_{net_i}) = g(\sum_j q_{w_{ij}}(q-1)_{y_i})$

Step 3: Calculating lost Q_{δ_i} at the output layer
$E = \frac{1}{2}\sum_{i=1}^{n}(d^k_i - Q_{y_i})^2 + E$
$Q_{\delta_i} = (d^k_i - Q_{y_i})g'(Q_{net_i})$

Step 4: Backpropagation
$\Delta q_{w_{ij}} = \mu q_{\sigma_i^{q-1} y_i}$
$q_{w^{new}_{ij}} = q_{w^{old}_{ij}} + \Delta q_{w_{ij}}$
$(q-1)_{\delta_i} = g'(q-1)_{net_i} \sum_j q_{w_{ij}} q_{\delta_j}$

Step 5: Checking repetition condition
if $k < p$ **then**
    $k = k + 1$
    Goto step 1
else
    Goto step 6
end if

Step 6: Checking learning lost
if $E < E_{max}$ **then**
    learning procedure stops and weighted parameters are provided
else
    $E = 0$
    $k = 1$
    Goto step 1 for another learning repetition
end if

space as V^{wrd}. The word embedding's dimension d^{wrd} is a hyperparameter to be manually selected by the user. Similar to [10], we denote $W \in \mathbb{R}^{d^{wrd} \times |V^{wrd}|}$ as an embedding matrix where each column $W_i \in \mathbb{R}^{d^{wrd}}$ corresponds to the word embedding of the i^{th} word in the word space. By r^{wrd}, we denote the embedding level of word and it is calculated by the following multiplication:

$$r^{wrd} = W v^w \tag{2}$$

where v^w is a vector of length $|V^{wrd}|$ which has the value of 1 is set at index w and 0 in other indexes. W^{wrd} is a parameter matrix to be determined.

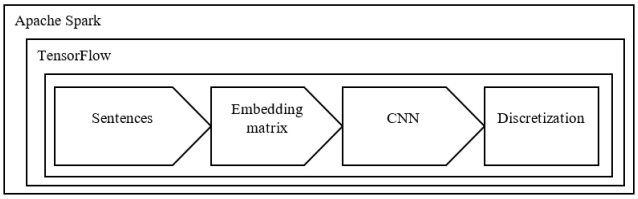

FIGURE 2. The design concepts of our proposed distributed framework for automating opinion discretization.

IV. OUR PROPOSED DISTRIBUTED FRAMEWORK FOR AUTOMATING OPINION DISCRETIZATION

Within this Section, the authors eventually introduce the design concepts that drive the development of the proposed framework, the core components and how they combine to solve text discretization task. The key components and implementation of our proposed big data processing system are the core of our contribution. Then, we describe the four-step procedure of constructing a word embedding matrix. We characterize our implementation of TensorFlow-based CNN. Finally, we present the data analysis and the model's training via Apache Spark.

A. DESIGN CONCEPTS

Deep learning has become one of the main research directions of artificial intelligence and machine learning. Specifically, deep learning (i) achieves high result accuracy in most of the very challenging domains, (ii) utilizes a huge velocity of information for supervised feature extraction, and (iii) avoids the expensive design of handcrafted features. Because of the dimensionality curse, training deep models on conventional computing systems, specifically in the context of text mining, is rather slow and needs anywhere from a couple of hours to some weeks to complete. As distributed programming becomes common, the composability of powerful deep learning models and the idea of distributed computing will be one of the most critical concerns for both performance and usability in the age of big data. Much of automating application is exploratory, with users willing to integrate quickly into a working pipeline. In this paper, the authors would like to discuss a scalable distributed framework over Apache Spark. This enables learning distribution using many computing nodes on a cluster where the continuously accessed data is cached to running memory, consequently accelerating up the updating of models' parameters by multiple folds. Time-effectiveness of opinion discretization is enabled by deploying multi-millions-parameters deep learning models to cope with the increased demand for adaptive opinion discretization systems. In another word, we introduce a Spark-based distributed framework and leverage it to build other machine learning tasks over it. The overview of our proposed distributed framework for automating opinion discretization from text corpora is presented in Figure (2).

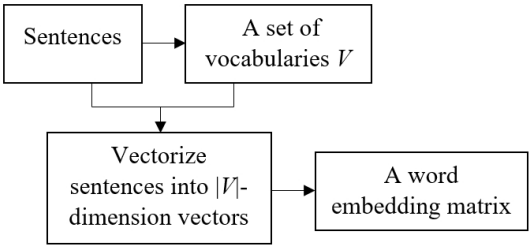

FIGURE 3. The formation of a word embedding matrix.

B. WORD EMBEDDING MATRIX

There are several methods to establish a word embedding matrix [49], [50]. However, in our work, we establish a word embedding matrix without concerning about sentence syntax and semantics. Figure (3) visualizes the formation of such word embedding matrix. The process of building the matrix is described as a four-step approach: (i) Building a set of vocabularies V from sentences, (ii) Replacing words in sentences by their indexes in V, (iii) Embedding words from sentences in step i, and (iv) Establishing a word embedding matrix.

C. TENSORFLOW-BASED CNN IMPLEMENTATION

The architecture can be broken down into several components for more details. The input data *input_x* is the first part. Embedding layer is the first layer of the architecture which is the projection of *input_x* into a set of vocabularies V. Regarding the convolution layer and max-pooling layer, the input data is the word embedding matrix. Activation function ReLU [26] is applied by default. This is done to help over-fitting by providing an abstracted form of the representation. Dropout layer is a regularization trick used to guide the network to distribute the learned representation across all the neurons. Basically, dropout randomly turns off a fraction of the nodes in the layer by setting their values to zero during training. The output layer is fully connected containing the number of prediction labels. Softmax is applied to predict the label with the highest probability. The cross-entropy is used for categorical data which is given by

$$E(u, v) = -\sum_{x} u(x) \log v(x). \tag{3}$$

For the training optimization, we use the adaptive learning rate Adam algorithm [18] as the optimization procedure. An example TensorFlow-based CNN implementation of a sentence "I love this phone!" is illustrated in Figure (4).

D. MODEL'S TRAINING VIA APACHE SPARK

The authors present the training procedure via Apache Spark in Figure (5). From the executive order, the authors systematically characterize that the training procedure contains two phases. Data analysis is the first phase and the second phase is model training.

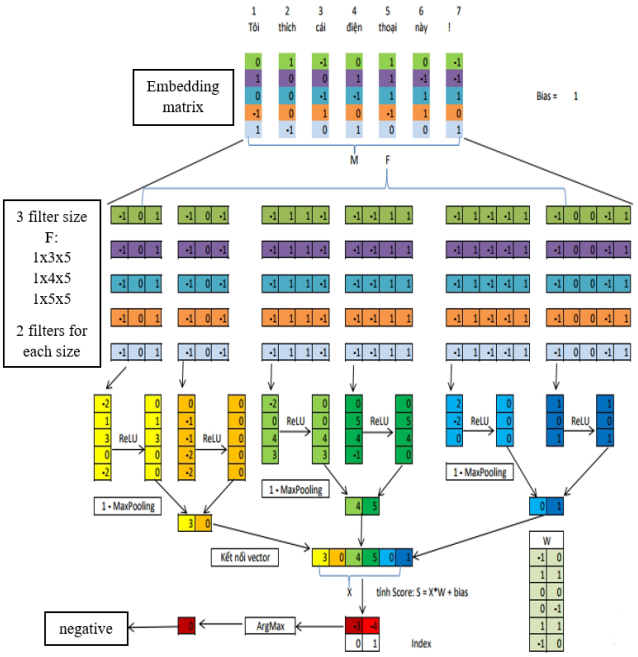

FIGURE 4. An example of TensorFlow-based CNN implementation of a sentence "I love this phone!" in Vietnamese. Note that the hyperparameters' values have not converged.

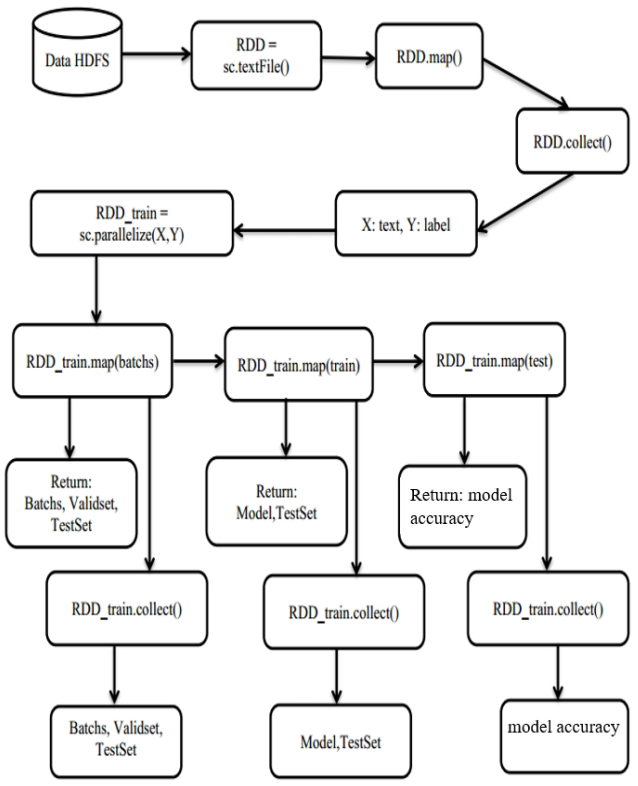

FIGURE 5. The procedure of distributed training CNN via Apache Spark.

1) DATA ANALYSIS PHASE

As already mentioned in the previous Sections, Apache Spark's main programming abstraction is RDDs, which are

the collections of data distributed across a computing platform that could be executed in a distributed or parallel fashion. At first, data from Hadoop distributed file system (HDFS) is transferred into resilient distributed datasets (RDD) by the function *sc.textFile()* and *RDD.map()*. Then, the returning data contains *X:text*, e.g. sentences, and *Y:label*, e.g. the corresponding label of *X:text*, by applying the function *RDD.collect()*.

2) MODEL TRAINING PHASE

The model training phase consists of several sub-phases. In the first sub-phase, e.g. Map-1, the output of data analysis from the previous data analysis phase is converted into RDD by the function *sc.parallelize(X,Y)*. Then the function *RDD_train.map(batchs)* randomly splits data into training, validation test batches. In the second sub-phase, e.g. Reduce-1, data returned from Map-1 containing batchs, validation set and test set by utilizing the function *RDD_train.collect()*. Mapa-2 is the third sub-phase where data returned from Map-1 is continuously trained by the function *RDD_train.map(train)*. The return of the function is a trained model and an equivalent test set. Then in Reduce-2 sub-phase, data returned from Map-2 containing the trained model and the equivalent test set. Next, in Map-3 sub-phase, data returned from Map-2 is continuously trained by the function RDD_train.map(test). The return of the function is the model's classification accuracy. Finally, in Reduce-3 sub-phase, data returning from Map-3 is the model's classification accuracy.

V. EXPERIMENT

In this Section, the authors present the dataset collection and labeling for a supervised machine learning task. Next, several dataset splitting schemes are considered. Finally, the authors present the experimental results and discussion.

A. DATASET

Experiment dataset is scrawled from Facebook comprising 4259 comments mentioning about opinions and thoughts on education subject. In an input dataset suitable for supervised learning tasks, a text corpus can have attached to it a set of labels which best describe the content of that corpus in a custom taxonomy used in that specific problem. The labels are manually assigned by five NLP experts and the majority vote is applied to decide 2192 negative and 2067 positive labels. Readers might find similar data collection and labeling procedure from the work done by [46].

Machine learning models have the fundamental goal of making accurate predictions on unseen instances beyond those appeared in the training set. To estimate the quality of models' predictions with data it has not seen, we can split a portion of the data for which we already know the answer as a proxy for the unseen data. Then we evaluate how well the model predicts for that data. Typically, training dataset contains observations used to fit a learning model. Validation dataset comprises instances used to provide an unprejudiced evaluation of the learning model by tuning hyperparameters. Test dataset includes samples of data used to provide an unbiased evaluation of the final learning model fit on the training dataset. The authors randomly shuffle the data into training, test, and validation sets without replacement in every experiment. In our experiment, we set up two different dataset splitting schemes by tuning various split ratios.

TABLE 1. Dataset splitting scheme 1.

Label	Samples	Training set	Validation set	Test set
0	2192	1776	197	219
1	2067	1675	186	206

TABLE 2. Dataset splitting scheme 2.

Label	Samples	Training set	Validation set	Test set
0	2192	1074	460	658
1	2067	1013	434	620

1) DATASET SPLITTING SCHEME 1

The proportion of pre-training and test portions is 90% and 10% respectively. Within 90% of the pre-training portion, we then split it into training and validation portions 90% and 10% respectively. The dataset splitting scheme 1 is presented in Table (1).

2) DATASET SPLITTING SCHEME 2

the proportion of pre-training and test portions is 70% and 30% respectively. Within 70% of the pre-training portion, we then split it into training and validation portions 70% and 30% respectively. The dataset splitting scheme 2 is presented in Table (2).

B. EXPERIMENT RESULTS

We describe the experimental results in the following two scenarios. While scenario 1 presents experimental results on dataset splitting scheme 1, scenario 2 shows experimental results on dataset splitting scheme 2. The computing infrastructure and virtual machines are set up on a normal desktop.

1) SCENARIO 1: EXPERIMENTAL RESULTS ON DATASET SPLITTING SCHEME 1

In this scenario, we evaluate our proposed distributed framework on dataset splitting scheme 1. The results are shown in Table (3). We randomly shuffle dataset without replacement 5 times and execute our model. Then we take an average in the end. The percentage of accuracy score is 72.85 ± 2.28 within the running time of 2836.7 ± 7.5 seconds or approximately 47.2 minutes including obtaining data, training model, distributing workload and reporting the result.

One considerable characteristic of Vietnamese is the signs of words, e.g. á à ạ ã ạ â ă. Therefore, we also investigate how our distributed framework handles this characteristic.

TABLE 3. Experimental results on dataset splitting scheme 1 in case of Vietnamese sentences without signs.

Experiment	# Sentences	# Correct Prediction	Percentage (%)	Running time (second)
1	425	320	75.29	2848.3
2	425	297	69.88	2830.7
3	425	301	70.82	2828.9
4	425	309	72.71	2833.1
5	425	321	75.53	2842.6
Average	**425**	**309.6**	$\mathbf{72.85 \pm 2.28}$	$\mathbf{2836.7 \pm 7.5}$

TABLE 4. Experimental results on dataset splitting scheme 1 in case of Vietnamese sentences with signs.

Experiment	# Sentences	# Correct Prediction	Percentage (%)	Running time (second)
1	425	315	74.12	2761.8
2	425	283	66.59	2809.4
3	425	310	72.94	2888.5
4	425	331	77.88	2868.8
5	425	312	73.41	2923.0
Average	**425**	**310.2**	$\mathbf{72.99 \pm 3.64}$	$\mathbf{2850.3 \pm 57.6}$

TABLE 5. Experimental results on dataset splitting scheme 2 in case of Vietnamese sentences without signs.

Experiment	# Sentences	# Correct Prediction	Percentage (%)	Running time (second)
1	1277	891	69.77	1730.2
2	1277	896	70.16	1724.9
3	1277	891	69.77	1715.8
4	1277	898	70.32	1701.0
5	1277	928	72.67	1724.5
Average	**1277**	**900.8**	$\mathbf{70.54 \pm 1.09}$	$\mathbf{1719.3 \pm 10.2}$

Table (4) presents its experimental results. The accuracy score is 72.99 ± 3.64 within the running time of 2850.3 ± 57.6 seconds or approximately 47.5 minutes. Observing a slight difference from the results in the case of sign-elimination, we can conclude that our framework performs well in the case of Vietnamese signs.

2) SCENARIO 2: EXPERIMENTAL RESULTS ON DATASET SPLITTING SCHEME 2

Similar to scenario 1 mentioned above, we also evaluate our proposed distributed framework on dataset splitting scheme 2. The results are shown in Table (5). We randomly shuffle dataset without replacement 5 times and execute our model. Then we take an average in the end. The percentage of accuracy score is 70.54 ± 1.09 within the running time of 1719.3 ± 10.2 seconds or approximately 28.6 minutes. We also investigate how our distributed framework handles Vietnamese signs on dataset splitting scheme 2. Table (6) presents its experimental results. The percentage of accuracy is 69.52 ± 1.14 within the running time of 1687.0 ± 72.1 seconds or approximately 28.1 minutes.

3) REMARKS

Overall, the experimental results reflect the difference between the two dataset splitting schemes. The prediction accuracy of scheme 2 is lower than that of scheme 1 as well as the running time of scheme 2 is higher than that of scheme 1. The reason is that the training and validation

TABLE 6. Experimental results on dataset splitting scheme 2 in case of Vietnamese sentences with signs.

Experiment	# Sentences	# Correct Prediction	Percentage (%)	Running time (second)
1	1277	880	68.91	1749.4
2	1277	864	67.66	1736.9
3	1277	905	70.87	1734.8
4	1277	899	70.40	1559.6
5	1277	891	69.77	1654.4
Average	**1277**	**887.8**	$\mathbf{69.52 \pm 1.14}$	$\mathbf{1687.0 \pm 72.1}$

portions are 90% in scheme 1 comparing with only 70% in scheme 2. There is almost no significant difference in term of prediction accuracy and running time of the framework's performance in case of Vietnamese signs and sign-elimination. Another important thing to note is that the standard deviation in the case of Vietnamese sentences with signs is about 7 times lower than that of Vietnamese sentences without signs. This phenomenon exists in two dataset splitting schemes.

VI. CONCLUSION

By conducting this research paper, the authors have investigated a Spark-based distributed convolutional neural networks and applied it on a critical task of natural language processing. We aim our research at proposing analytics for opinion discretization on Facebook that has not been studied in the literature, especially Vietnamese text sources. The potential generalization of deep learning and distributed computing in the context of text mining has been highlighted. An application in Vietnamese natural language processing has been deployed by utilizing cutting-edge machine learning frameworks and big-data technologies that would indicate blossom research in this direction. The development of the framework started with a specific focus on binary text discretization, but soon expanded toward many other text-categorization-based problems, distributed language modeling and quantification. We achieve several vital results on both theory and practice such as design concepts, framework components, convolution, neural network architecture, a configuration of CNN, implementation of CNN on open source software, investigation of an un-studied discretization task. We envision that the proposed distributed framework will become increasingly effective as more data have been generated on the Web.

REFERENCES

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, and M. Kudlur, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, 2016, pp. 265–283.
[2] A. Das and S. Bandyopadhyay, "Topic-based Bengali opinion summarization," in *Proc. 23rd Int. Conf. Comput. Linguistics, Posters. Assoc. Comput. Linguistics*, 2010, pp. 232–240.
[3] A. Krenker, J. Bešter, and A. Kos, "Introduction to the artificial neural networks," in *Artificial Neural Networks—Methodological Advances and Biomedical Applications*, K. Suzuki, Ed. IntechOpen, 2011. doi: 10.5772/15751.
[4] A. Bossard, M. Généreux, and T. Poibeau, "CBSEAS, a summarization system integration of opinion mining techniques to summarize blogs," in *Proc. 12th Conf. Eur. Assoc. Comput. Linguistics, Demonstration Session*, 2009, pp. 5–8.

[5] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, Feb. 2003.

[6] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Found. Trends Inf. Retr.*, vol. 2, nos. 1–2, pp. 1–135, 2008.

[7] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, Aug. 2011.

[8] G. Daytona. *Sort Benchmark: Home Page of Sort Benchmarks*. Accessed: Sep. 27, 2016. [Online]. Available: http://sortbenchmark.org/

[9] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[10] C. D. Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proc. 25th Int. Conf. Comput. Linguistics, Tech. Papers (COLING)*, 2014, pp. 69–78.

[11] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[12] M. Hu and B. Liu, "Opinion extraction and summarization on the Web," in *Proc. AAAI*, vol. 2, 2006, pp. 1621–1624.

[13] J. G. Conrad, J. L. Leidner, F. Schilder, and R. Kondadadi, "Query-based opinion summarization for legal blog entries," in *Proc. 12th Int. Conf. Artif. Intell. Law*, 2009, pp. 167–176.

[14] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," 2014, *arXiv:1404.2188*. [Online]. Available: https://arxiv.org/abs/1404.2188

[15] H. Karau, *Learning Spark: Lightning-Fast Big Data Analysis*. Newton, MA, USA: O'Reilly Media, Inc., 2015.

[16] K. Lerman, S. Blair-Goldensohn, and R. McDonald, "Sentiment summarization: Evaluating and learning user preferences," in *Proc. 12th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2009, pp. 514–552.

[17] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*. [Online]. Available: https://arxiv.org/abs/1408.5882

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[21] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A Y. Ng, "On optimization methods for deep learning," in *Proc. 28th Int. Conf. Int. Conf. Mach. Learn.* Bellevue, WA, USA: Omnipress, 2011, pp. 265–272.

[22] Y. Li, L. Xu, F. Tian, L. Jiang, X. Zhong, and E. Chen, "Word embedding revisited: A new representation learning and explicit matrix factorization perspective," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, 2015, pp. 3650–3656.

[23] F. McSherry, M. Isard, and D. G. Murray, "Scalability! But at what cost?" in *Proc. 15th Workshop Hot Topics Oper. Syst. (HotOS)*, 2015, p. 14.

[24] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[25] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 168–177.

[26] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.

[27] M. M. Nelson and W. T. Illingworth, *A Practical Guide to Neural Nets*. Reading, MA, USA: Addison-Wesley, 1991.

[28] N. Nodarakis, S. Sioutas, A. K. Tsakalidis, and G. Tzimas, "Large scale sentiment analysis on Twitter with spark," in *Proc. EDBT/ICDT Workshops*, 2016, pp. 1–8.

[29] N. T. Duyen, N. X. Bach, and T. M. Phuong, "An empirical study on sentiment analysis for vietnamese," in *Proc. Int. Conf. Adv. Technol. Commun. (ATC)*, Oct. 2014, pp. 309–314.

[30] P. Zikopoulos and C. Eaton, *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*. New York, NY, USA: McGraw-Hill, 2011.

[31] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: Sentiment classification using machine learning techniques," in *Proc. ACL Conf. Empirical Methods Natural Lang. Process.*, Association for Computational Linguistics, vol. 10, 2002, pp. 79–86.

[32] A. Ortigosa, J. M. Martín, and R. M. Carro, "Sentiment analysis in Facebook and its application to e-learning," *Comput. Hum. Behav.*, vol. 31, pp. 527–541, Feb. 2014.

[33] Q.-T. Ha, T.-T. Vu, H.-T. Pham, and C.-T. Luu, "An upgrading feature-based opinion mining model on Vietnamese product reviews," in *Proc. Int. Conf. Active Media Technol.* Berlin, Germany: Springer, 2011.

[34] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 1631–1642.

[35] D. E. Rumelhart, J. L. McClelland, and PDP Research Group, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Psychological and Biological Models*, vol. 2. 1986.

[36] A. Severyn and A. Moschitti, "Twitter sentiment analysis with deep convolutional neural networks," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2015, pp. 959–962.

[37] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "Learning semantic representations using convolutional neural networks for Web search," in *Proc. 23rd Int. Conf. World Wide Web*, 2014, pp. 373–374.

[38] The Apache software Foundation. *Apache Spark*. Accessed: Sep. 27, 2016. [Online]. Available: http://spark.apache.org/

[39] S. Kumar and D. Chatterjee, "IIT Kharagpur at TAC 2008: Statistical model for opinion summarization," 2008.

[40] T.-T. Vu, H.-T. Pham, C.-T. Luu, and Q.-T. Ha, "A feature-based opinion mining model on product reviews in Vietnamese," in *Semantic Methods for Knowledge Management and Communication*. Berlin, Germany: Springer, 2011, pp. 23–33.

[41] W.-T. Yih, K. Toutanova, J. C. Platt, and C. Meek, "Learning discriminative projections for text similarity measures," in *Proc. 15th Conf. Comput. Natural Lang. Learn. Assoc. Comput. Linguistics*, 2011, pp. 247–256.

[42] W.-T. Yih, X. He, and C. Meek, "Semantic parsing for single-relation question answering," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, vol. 2, 2014, pp. 643–648.

[43] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found. Trends Signal Process.*, vol. 7, nos. 3–4, pp. 197–387, 2014.

[44] S. Chintala. *Convolutional Network Benchmarks*. Accessed: Sep. 27, 2016. [Online]. Available: http://www.github.com/soumith/convnet-benchmarks

[45] Yahoo Tensorflow. *Tensorflow on Spark*. Accessed: Sep. 21, 2017. [Online]. Available: http://www.github.com/yahoo/tensorflowonspark

[46] L. Yang, J. Shi, B. Chern, and A. Feng. *Open Sourcing TensorFlowOnSpark: Distributed Deep Learning on Big-Data Clusters*. Accessed: Sep. 21, 2017. [Online]. Available: http://yahoohadoop.tumblr.com/post/157196317141/open-sourcing-tensorflowonspark-distributed-deep

[47] B. Liu, *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge, U.K.: Cambridge Univ. Press, 2015.

[48] J. Bian, B. Gao, and T.-Y. Liu, "Knowledge-powered deep learning for word embedding," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Berlin, Germany: Springer, 2014.

[49] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2177–2185.

[50] T. Shi, Z. Liu, Y. Liu, and M. Sun, "Learning cross-lingual word embeddings via matrix co-factorization," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, vol. 2, 2015, pp. 567–572.

[51] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion* vol. 42, pp. 146–157, Jul. 2018.

[52] Z. Qin, F. Yu, C. Liu, and X. Chen, "How convolutional neural network see the world—A survey of convolutional neural network visualization methods," 2018, *arXiv:1804.11191*. [Online]. Available: https://arxiv.org/abs/1804.11191

[53] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[54] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica, "Apache spark: A unified engine for big data processing," *Commun. ACM*, vol. 59, no. 11, pp. 56–65, 2016.

[55] N. Duong-Trung, *Social Media Learning: Novel Text Analytics for Geolocation and Topic Modeling*. Göttingen, Germany: Cuvillier Verlag, 2017.
[56] R. Zafarani, M. A. Abbasi, and H. Liu, *Social Media Mining: An Introduction*. Cambridge, U.K.: Cambridge Univ. Press, 2014.
[57] C. C. Aggarwal and C. Zhai, Eds., *Mining Text Data*. Springer, 2012.
[58] R. Ignazio, *Mesos in Action*. Shelter Island, NY, USA: Manning Publications, 2016.
[59] N. Duong-Trung, L.-D. Quach, and C.-N. Nguyen, "Learning deep transferability for several agricultural classification problems," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 1, pp. 58–67, 2019.
[60] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.
[61] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to WordNet: An on-line lexical database," *Int. J. Lexicogr.*, vol. 3, no. 4, pp. 235–244, 1990.

HIEP XUAN HUYNH received the Engineer degree from Can Tho University (CTU), the master's degree from the l'Institut de la Franco phonie pour l'Informatique (IFI), the Ph.D. degree from the Polytechnics School of Nantes University (Polytech'Nantes), and the Habilitation à Diriger des Recherches (HDR) degree from the l'Université de Bretagne Occidentale (UBO), all in computer science. He is an Associate Professor of computer science with the College of Information and Communication Technology, Can Tho University, Vietnam. His research interests include modeling of decisions with interestingness measure and fuzzy integral, recommender system with statistical implicative analysis, a cyber-physical system with cellular automata and wireless sensor networks, metrics for computer vision and population dynamics, agriculture, aquaculture, and environment issues.

VU TUAN NGUYEN received the B.Sc. and M.Sc. degrees in computer science from Can Tho University, Vietnam, where he is currently pursuing the master's degree in computer science with the College of Information and Communication Technology. His research interests include modeling of decisions with interestingness measure and fuzzy integral, recommender system with statistical implicative analysis, a cyber-physical system with cellular automata and wireless sensor networks, metrics for computer vision, and population dynamics, agriculture, aquaculture, and environment issues.

NGHIA DUONG-TRUNG received the Ph.D. degree in machine learning from Information Systems and Machine Learning Lab (ISMLL), Hildesheim University, Germany, in 2017. He works as a full-time Lecturer with Can Tho University of Technology and a Visiting Lecturer with FPT University, which is the best private university in Vietnam established by FPT Corporation. He has contributed several research papers in top-tier conferences and journals. He has also published a textbook titled "Social Media Learning: Novel Text Analytics for Geolocation and Topic Modeling" to address some encouraging subjects in social media such as text-based geolocation and topic modeling, and to solve them by developing robust machine learning algorithms. His research interest includes machine learning, development and application of deep learning for multiple data sources. His grants and awards include the Ph.D. Fellowship (ISMLL and Ministry of Education and Training of Vietnam), the Teaching Assistant Grant (ISMLL), the Best ECML/PKDD Discovery Challenge Award, and the Best Presentation Award (ICMLSC).

VAN-HUY PHAM received the M.Sc. degree in computer science from the University of Sciences, Ho Chi Minh City, Vietnam, in 2007, and the Ph.D. degree in computer science from Ulsan University, South Korea, in 2015. Since then, he has been a Lecturer and Researcher with the Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh City, Vietnam. His current research interests include artificial intelligence, image processing, and computer vision.

CANG THUONG PHAN received the B.Sc. degree in informatics engineering from Can Tho University, Can Tho, Vietnam, in 1998, the M.Sc. degree in computer science from the Asian Institute of Technology, Bangkok, Thailand, in 2006, and the Ph.D. degree in informatics under the supervision of Professor Philippe Rigaux and Professor Laurent d'Orazio from Blaise Pascal University, Clermont-Ferrand, France, in 2014, and he held a postdoctoral position with the LARIDEPED Laboratory, Université du Québec à Trois-Rivières, Canada, in 2015. He has been a Senior Lecturer with the College of Information Technology, since 1998, and the Head of the Mobile Networks and Big Data Lab, since 2016, Can Tho University, Vietnam. He is the author of a recently remarkable article as "Towards a Service-Oriented Architecture for Knowledge Management in Big Data Era" (IGI Publishing, 2018). His research interests include algorithmic foundations of massive data and big data-driven knowledge management systems, complex large-to-large join algorithms, probabilistic data structures, machine learning, and service-oriented architecture for medical knowledge management in the big data era. He participated as an Associate Editor for the track "Knowledge-intensive smart services and their applications" of the conference ECIS 2018 (European Conference on Information Systems). His awards include the Postdoctoral Fellowship with LARIDEPED, Université du Québec à Trois-Rivières, Canada (Project 2014-NP-173669 of the FRQSC), and the Master and Ph.D. competitive scholarship from the Ministry of Education and Training of Vietnam (Project 322- training overseas human resources with Government's budget).