Automatic Sentiment Detection Of Tweets

Loyiso Jiya, Mehrdad Ghaziasgar
Department of Computer Science, University of the Western Cape, Private Bag X17, Cape Town, Bellville, 7535
Tel: (+27) 21 959 3010, Fax: (+27) 21 959 3006
3338868@myuwc.ac.za, mghaziasgar@uwc.ac.za

***Abstract*—Sentiment detection is a field of natural language processing that aims to automatically detect the overall emotion expressed in a piece of text on a given scale. One way of doing this is to determine whether the emotion expressed in the text is positive, neutral or negative. The large amounts of text data generated in social media everyday has resulted in a data deluge. This data can be leveraged to gauge public opinion on a range of topics. Therefore, automatic sentiment detection has become an increasingly important area of research. This research proposes the use of Extreme Gradient Boosting and a Residual Neural Network coupled with Information gain and/or Principal Components Analysis, for sentiment detection in Tweets.**

***Index Terms*—Sentiment, Detection, Analysis, Emotion, Deep, Learning, Support Vector Machines**

I. Introduction

Sentiment detection is a niche subarea of text classification that emerged as a new research area in the early 2000s due to the large increase of subjective texts in social media, forums, blogs etc. [1]. It aims to use of state-of-the-art machine learning (ML) techniques towards automatically determining the emotion/sentiment expressed in a piece of text on a given—usually discrete—scale. The sentiment scale used can range from a fundamental classification into {Positive, Negative, Neutral} categories, to much more sophisticated scales that represent varying degrees of positive and negative sentiment. Sentiment detection is also commonly referred to as sentiment analysis, opinion mining, review mining, subjectivity extraction, and opinion extraction in the literature [2]. In this paper, we will continue to refer to it as sentiment detection.

Social media platforms contain a wealth of free, readily available data that can be gathered, analyzed and used to gauge the public perception of a particular subject. Twitter is one such platform, that has become one of the biggest social media platforms. As of 2018, Twitter had approximately 300 million active users per month. Users readily share their opinions about a variety of subjects continuously throughout each day.

By leveraging the data on social media, the applications of sentiment detection are growing with the times. Recently, sentiment detection has been applied to mental illness detection[3], sarcasm detection [4], movie review sentiment detection [5], among other applications.

One particularly useful and lucrative application of sentiment detection is towards business intelligence, specifically towards automatically gauging the public perception of products offered by companies. Companies require feedback from their customers to remain relevant and competitive. Traditionally, this was achieved by means of manually gathered customer reviews, interviews, focus groups etc which was time-consuming, cumbersome and costly.

Sentiment detection holds the promise of being able to gauge the public perception on a near-real-time basis, at significantly lower cost and with much greater ease. Large companies like Amazon and Dell, among many others, are already leveraging this technique to evaluate customers' reactions to their services [6].

Sentiment detection is clearly a useful and important area of research. This research aims to devise a sentiment detection system for Tweets. The rest of the paper is organized as follows: Section II provides a review of relevant recent literature into sentiment detection which is used to inform the focus of this research which is detailed in Section III.

II. Literature Review

This section provides a brief summary of recent relevant research in the field of sentiment detection.

Contratres et al. [7] attempted to remedy the problem of making relevant recommendations to new users in recommender systems for e-commerce websites [7]. They proposed the use of sentiment detection on the social media profile that a new user of an e-commerce website volunteers to give to the system to help generate more relevant user-specific product recommendations. They compared a naive Bayes and Support Vector Machine (SVM) classifier, coupled with term frequency-inverse document frequency (TF-IDF) towards sentiment detection. In their future work, they proposed using deep learning techniques to classify product categories and sentiment.

Zheng et al. [8], focused on the feature extraction techniques that can be utilized in sentiment detection. The paper mainly focused on sentiment detection of Chinese online reviews using a SVM for classification. They compared two feature extraction techniques, namely, N-Part-of-Speech-grams (N-PoS-grams) and TF-IDF. In the end, they found that TF-IDF outperformed the different combinations of N-PoS-grams.

Jiaqiang and Xialoin [9] compared a range of text pre-processing alternatives and studied their effects on sentiment detection on tweets, in addition to comparing random forests (RFs), SVMs, logistic regression and naive Bayes classifiers to perform sentiment detection, and N-PoS-grams and prior polarity feature extraction. They observed that RFs and SVMs are more sensitive to different pre-processing techniques.

Lauren et al. [10], expanded on their previous work where they proposed an Extreme Learning Machine (ELM) for word embeddings. They applied their previously designed ELM-based word embedding for sentiment detection and sequence labeling. Their implementation achieved better results than the standard word2vec word embedding and global vectors (GloVec) model in both sentiment detection and sequence labelling.

Al-Smadi et al. [11], compared deep learning techniques to a SVM in sentiment detection for Arabic hotel reviews. In this study, Arabic Natural Language Processing was used for pre-processing, and N-PoS-grams tagging and word embeddings were used for feature extraction. The SVM outperformed the different deep learning techniques for sentiment detection.

In another study [12], a deep convolutional neural network (CNN) was used in an ensemble algorithm to achieve better results. The study compared the proposed (ensemble) model with other variations of a CNN and found that the proposed model achieved better results than some CNNs. The system used global vectors, the word2vec model, TF-IDF, and bag-of-words for feature extraction.

III. Discussion and Research Focus

Given the surveyed literature above, it is evident that many of the studies conducted are comparative in nature. In the same way, the proposed research will be a comparison of different feature extraction methods; similar to Zheng et al. [8], TF-IDF will be used for feature extraction. However, unlike Zheng et al. [8], the TF-IDF will be compared against the ELM word embeddings. Moreover, for feature selection, Zainuddin et al. [13]. showed the utility of PCA as a feature selection method. Therefore, the PCA method will be compared to IG for feature selection.

Furthermore, most of the papers surveyed suggested implementing deep learning approaches in their future work, but SVMs were found to out-perform deep learning approaches in many cases. Thus, in this paper we propose a ResNet as our deep learning approach which has not been used in the surveyed literature. Finally, in addition to the ResNet, we propose using the XGBoost algorithm. Both of these techniques will be compared to other ML techniques such as SVMs and naive Bayes.

Figure 1 depicts the proposed system architecture. Referring to the figure, the Stanford Twitter Sentiment Corpus [14] will be pre-processed, and features will be selected and extracted from it. This will be followed by training in the (top) training phase of the aforementioned ML techniques using a subset of the data set. Finally, the remainder of the data set will be used to test the proposed approach in the (bottom) testing phase.

IV. Proposed Implementation

A. Data Set

For this paper we will be using the Stanford Twitter Sentiment Corpus (Sentiment140) *Stanford Twitter Corpus* (STS)[14]. The data was created by Alec Go, Richa Bhayani, and Lei Huang, who were Computer Science graduate students at Stanford University. On their website as referenced above, the use cases stated include: brand management, polling and planning a purchase. The file consists of six fields:

- the polarity of the tweet (e.g. 0 = negative, 2 = neutral, 4 = positive)
- the id of the tweet (e.g. 2087)
- the user that tweeted (e.g. robotickilldozr)
- the text of the tweet (e.g. Lyx is cool)

In this research, the data will be divided into training and testing portions as illustrated in Fig. 1.

B. Pre-processing

For any machine learning algorithm to work for any NLP task the text needs to be represent as numerical data for statistical analysis. However, it is unlikely for most text to immediately be ready ready for feature extraction. Because, if this is the case, the feature extraction will generate a lot of features that are not important to the task at hand and thereby generate noise which will affect the final model's accuracy and its ability to generalize. Therefore, to alleviate the noise in text data, the data must first go through the stages of pre-processing.

1) Tokenization: Tokenization is the process of breaking down continuous text into words, phrases, and removing spaces from a given text. The resulting words, phrases, and or symbols are referred to as tokens[15].

2) Stop words removal: In the context of social media, text and document classification is often fill with common words like 'an', 'this', 'that', 'is' that often carry a small weight relative to other words. These words are known as stop words. Most *natural language toolkits* (NLTK) have a list of stop words to be used during the pre-processing phase. This list is then used to reliminate the words from text or documents this consequently decrease the resulting features.

3) Stemming: Stemming transforms a word to obtain its variants using different linguistic processes like affixation, i.e. adding affixes. The most frequently used stemmer is the Porter Stemmer, this paper will also use this stemmer.

4) Spelling correction: With social media posts, there will often be words that are deliberately misspelled to achieve some sort of emphasis. For instance, someone would write 'Amazzzziiiiing', instead of just amazing. These are some of the spelling mistakes that one must deal with during pre-processing. Also, these are not the only spelling mistakes that are there but these are the most prevalent in our context. To mitigate this, we will use the python *pattern* library to correct the spelling mistakes contained in our text sample.

5) Noise removal: In the case of social media or blog posts, there can be a lot of symbols and characters that add no value to a machine learning model, but are rather relevant in the context of whatever social media platform the post is from. From example the hashtag *'#'* and the tag *'@'* characters on Twitter. Noise removal is performed as an attempt to remove some of these characters. However, because this technique requires the use of regular expressions, it is done

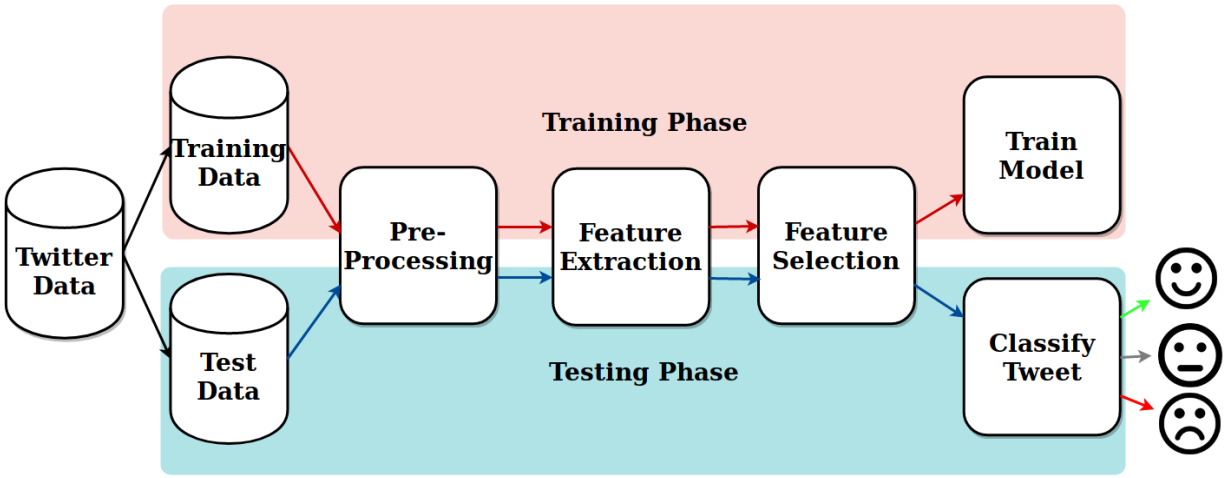

Fig. 1: Proposed sentiment detection system architecture.

at the potential expense of losing punctuation marks which are critical in understanding sentences.

V. Implementation

This section will focus on the machine learning techniques used in this paper, the experimental environment, training, and optimization of the machine learning models.

1) Extreme Gradient Boosting: The concept of "Gradient Boosting" originates from the paper *Greedy Function Approximation: A Gradient Boosting Machine* written by Jerome Friedman. Much of his work forms part of the Extreme Gradient Boosting algorithm. The algorithm is used for supervised learning problems. The model has parameters which must be learned from the data and tuned for better performance.

Not all the parameters were tuned for the implementation in this paper. The parameters that were tuned in our experiment are the *max_depth* which ensures the tree does not exceed a certain depth, the deeper a tree, the more complex it becomes. Other parameters include the number of trees in the whole tree and the sample of the data being used.

2) Residual Neural Network: In deep learning, there is a constant dilemma of adding more layers to potentially boost performance and keeping the the neural network from growing too big that some neurons start vanishing [16]. In response, different architectures of the residual neural network have been proposed and implemented this leading to the development of very deep neural networks that actually add value as they grow in size.

The ResNet consists of an "identity connection/layer" that is able to skip one or more layers as illustrated in Fig. 1. This then results in the ability for researcher to stack layers on a neural network. He et al. [17] in their paper proposed a pre-activation of residual blocks. Now this has resulted in a few arrangements of the architecture. The paper will use the original architecture of the ResNet.

3) Environment: The experimental setup for this research was a computer running Windows operating system with the following specifications and software:

- 16 Gigabytes of RAM
- Nvidia GeForce GTX 1060 graphics card with 6 Gigabytes of memory
- Hard Disk Drive with 250 Gigabytes
- Jupyter Notebook

4) Pre-processing: First, as suggested in the literature review section of this paper, the first step for any NLP experiment is data cleaning. For tokenization, WordPunctTokenizer was used, as it has been found to perform better than others. The NLTK comes standard with lists containing stopwords, lemmatized words, and stemmed word. These were used in the text pre-processing stage with the help of regular expressions to remove unwanted punctuation marks and characters. Furthermore, TF-IDF was used for feature extraction which resulted in a numeric vector which was subsequently used for the XGBoost classification algorithm.

The TF-IDF features depend on the input data. A large data set will result in a large number of features. For this experiment the data set has 250 000 instances which resulted in 47 781 features after feature extraction using TF-IDF.

5) Initial Training and Testing: Preliminary training and testing was performed on the chosen machine learning methods. Before optimization, an out of the box XGBoost classifier with the addition of the **tree_method** parameter which is used by passing the argument 'gpu_hist' to activate GPU computation. The F1 score for this classifier was 70% as seen in Table 1. The purpose of this initial test was to position ourselves for the next stage which is optimization and parameter tuning.

6) Training and optimization: Training and optimization was the most time consuming and arduous phase of the research. First, one has to understand the parameters of the

TABLE I: Baseline Tests for Desired Models

	F1-score (%)		
Prediction	XGBoost	ResNet(Adam Optimizer)	ResNet-10(Gradient Descent Optimizer)
negative	45	67	0
positive	70	72	67
overall	70	72	67

TABLE II: Improved results Desired Models

	F1-score (%)		
Prediction	XGBoost	ResNet(Adam Optimizer)	ResNet-10(Gradient Descent Optimizer)
negative	78	71	74
positive	73	80	76
overall	70	75	75

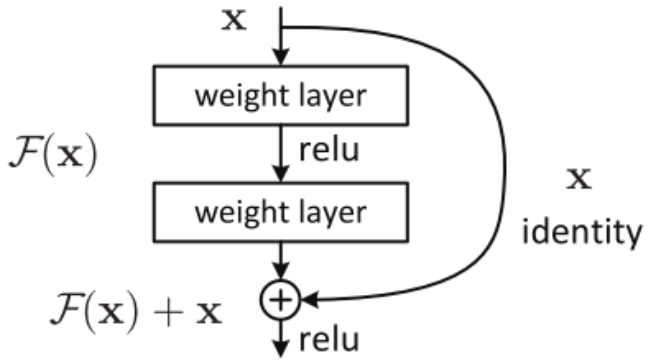

Fig. 2: Identity layer

model/machine learning algorithm that will be used. Then gradually narrow down which parameters will be focused on tuning the hyper parameters. Nonetheless, there are online forums and other literature that advise on which parameters to focus on like the one relevant to my case.

The Extreme Gradient Boosting algorithm is not computationally intense. Therefore, a few parameters that make significant impact to the overall prediction needed to be tuned. In evaluating a Random Forest Classifier, Foster tuned three parameters, namely the number of trees/estimators, depth of the forest, and the number of sample in each branch has [18]. Similarly, I performed a grid search of some parameters in an attempt to get the optimal parameters for the problem at hand.

Fig. 3 is a partial illustration of the grid search, as the search space was large and had to be fragmented to make it manageable, all the while updating the model parameters that are seemingly optimal, that is the ones that do not change.

For the Residual Neural Network, no grid search has been performed yet. First step was picking an optimizer that not only converges, but also performs well. The options were the Adam Optimizer and the Gradient Descent Optimizer. Gradient descent clearly converges. However, due to its adaptive learning rate, the Adam optimizer's performance fluctuates. However, after 12 epochs it appears to be beginning to stabilize, as seen in Fig. 4. Therefore, there is a need to train for longer epochs for a better model.

VI. Results and Analysis

In evaluating the models, the data set was split into 160 000 for training, 50 000 for testing and 40 000 for cross validation. In addition to that, 2 500 samples of the Semeval, of which some where neutral. A caveat of this project is that none of our models cater for neutral classes thus far. From the relatively small sample, the XGBoost model achieved 93% accuracy.

On the other hand, after cross validation and parameter tuning, the baseline F1-score as presented in Table 1 have also improved. Moreover, as compared to the previous table, there has been an improvement on the per class F1-score together with the overall F1 Scores. The XGBoost model has outperforms our deep learning technique by 7% when only looking at per-class f1 score..

VII. Conclusion

In conclusion, we proposed implementing two ML techniques to perform sentiment detection. That is, using Extreme Gradient Boosting algorithm and any arrangement of the ResNet architecture. In our experiments we have discovered that it is rather difficult to cross to the 80% F1-score when working with these two algorithms. For future work, we propose applying some kind of informed dimensionality reduction [19] feature selection and try a different architecture that applies the ResNes paradigm like the one used to classify web pages[20].

References

[1] R. Xia, C. Zong, and S. Li, "Ensemble of feature sets and classification algorithms for sentiment classification," *Information Sciences*, vol. 181, no. 6, pp. 1138–1152, 2011.

[2] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[3] S. C. Guntuku, D. B. Yaden, M. L. Kern, L. H. Ungar, and J. C. Eichstaedt, "Detecting depression and mental illness on social media: an integrative review," *Current Opinion in Behavioral Sciences*, vol. 18, pp. 43–49, 2017.

[4] A. C. Pandey, S. R. Seth, and M. Varshney, "Sarcasm detection of Amazon Alexa sample set," in *Advances in Signal Processing and Communication*. Springer, 2019, pp. 559–564.

[5] K. Chakraborty, S. Bhattacharyya, R. Bag, and A. E. Hassanien, "Comparative sentiment analysis on a set of movie reviews using deep learning approach," in *International Conference on Advanced Machine Learning Technologies and Applications*. Springer, 2018, pp. 311–318.

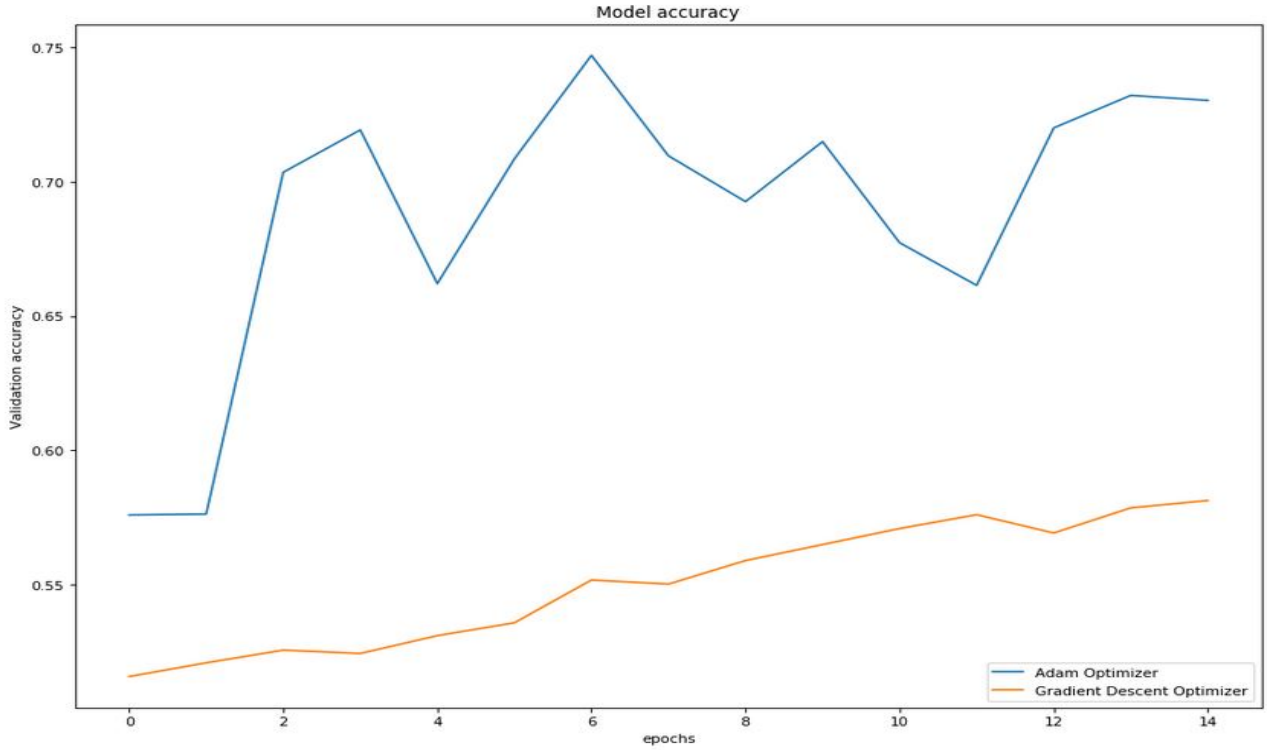

Fig. 3: Validation accuracy for deep learning model from different optimizers.

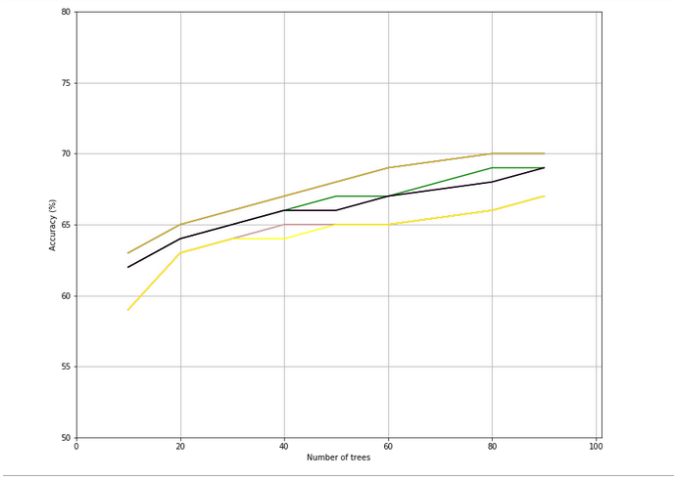

Fig. 4: Optimization of XGB: Cross-validation accuracy by tuning number of trees and depth of trees

[6] M. Oussalah, F. Bhat, K. Challis, and T. Schnier, "A software architecture for twitter collection, search and geolocation services," *Knowledge-Based Systems*, vol. 37, pp. 105–120, 2013.

[7] F. G. Contratres, S. N. Alves-Souza, L. V. L. Filgueiras, and L. S. DeSouza, "Sentiment analysis of social network data for cold-start relief in recommender systems," in *World Conference on Information Systems and Technologies*. Springer, 2018, pp. 122–132.

[8] L. Zheng, H. Wang, and S. Gao, "Sentimental feature selection for sentiment analysis of chinese online reviews," *International Journal of Machine Learning and Cybernetics*, vol. 9, no. 1, pp. 75–84, 2018.

[9] Z. Jianqiang and G. Xiaolin, "Comparison research on text pre-processing methods on twitter sentiment analysis," *IEEE Access*, vol. 5, pp. 2870–2879, 2017.

[10] P. Lauren, G. Qu, J. Yang, P. Watta, G.-B. Huang, and A. Lendasse, "Generating word embeddings from an extreme learning machine for sentiment analysis and sequence labeling tasks," *Cognitive Computation*, vol. 10, no. 4, pp. 625–638, 2018.

[11] M. Al-Smadi, O. Qawasmeh, M. Al-Ayyoub, Y. Jararweh, and B. Gupta, "Deep recurrent neural network vs. support vector machine for aspect-based sentiment analysis of arabic hotels reviews," *Journal of Computational Science*, vol. 27, pp. 386–393, 2018.

[12] M. Hanafy, M. I. Khalil, and H. M. Abbas, "Combining classical and deep learning methods for twitter sentiment analysis," in *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer, 2018, pp. 281–292.

[13] N. Zainuddin, A. Selamat, and R. Ibrahim, "Hybrid sentiment classification on twitter aspect-based sentiment analysis," *Applied Intelligence*, pp. 1–15, 2018.

[14] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," *CS224N Project Report, Stanford*, vol. 1, no. 12, p. 2009, 2009.

[15] L. M. Abualigah and A. T. Khader, "Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering," *The Journal of Supercomputing*, vol. 73, no. 11, pp. 4773–4795, 2017.

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer vision*. Springer, 2016, pp. 630–645.

[18] R. Foster, "A comparison of machine learning techniques for hand shape recognition," Master's thesis, University of the Western Cape, 2015.

[19] X. Bai, X. Gao, and B. Xue, "Particle swarm optimization based two-stage feature selection in text mining," in *2018 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2018, pp. 1–8.

[20] Y. Lin, "Rnn-enhanced deep residual neural networks for web page classification," Ph.D. dissertation, University of Calgary, 2016.