Utilizing a deep learning approach for detecting spam.

Venkatesh V Ankola
Department of Computer Science and Engineering, RV University
venkateshva.btech22@rvu.edu.in

Sushan Rai
Department of Computer Science and Engineering, RV University
sushanr.btech22@rvu.edu.in

Vasanth J
Department of Computer Science and Engineering, RV University
vasanthj.btech22@rvu.edu.in

Kalp Jain
Department of Computer Science and Engineering, RV University
kalpj.btech22@rvu.edu.in

Abstract

In the realm of the machine learning world, we come across different challenges, one of them is optimization of spam detection techniques. We come across different methodologies used to solve spam detection such as rule-based architecture, traditional ML algorithms and neural networks where we comprehensively study their traits. We then talk about DataSet and model architecture based on design principles that match our optimized approach to solve this problem of spam detection. We then talk extensively about each layer and its importance. Furthermore we deep dive into comparing traditional Machine learning algorithms with our feed-forward neural network to classify accurately if a message is a ham or a spam. In conclusion we find out that model architecture gravely improves the model performance.

1. Introduction

1.1 Problem Statement

In today's Era of digitalization, there has been a significant increase in usage of communication and networking in social media platforms,leading to an increase in specialized spam messages which have been difficult to detect. Spam being diverse and dangerous with a wide range of uses in advertisements to phishing money has not only increased the amount of spam messages but also gravely attacks the safeness of communications and networks. There has been an exponential increase of dependency on social media platforms for networking with personal and professional communities which has called for the need of efficient spam detection.

As mentioned above about the diverse use cases of spam messages, the report involves establishing a solution to the problem as well as reducing the challenges of detecting spam in future years to come. At a top level view, spam detection methodology often deals with recognizing the patterns in the text and classifying those features as a ham or a spam,with a usage of varieties of rules in statistics, calculus and machine learning. Although There is an inflation in the technology developments and increment in detailed techniques of detection, spam always prevails by escaping its routes through traditional methods of recognition, adapting itself to not being able to recognize.

Although there are spam generators advanced in technology, this report fights back against these challenges and sets itself in a journey where its has a conquest of determining those patterns to be able to win this everlasting battle.The report is like sherlock holmes trying to find patterns to solve a case against Spam generator which is a vicious murderer.

With increase in evolution and developments in recent years with artificial intelligence coming into light we look upon evergrowing and fast trends in development which has prompted us to go in the direction of advancing spam detection methods. This report strives to go in detail with the evolution in technology, machine learning and natural language processing to check their potential to change the waters of spam detection and produce a strong barrier between spam generation and spam detection techniques.

By ridiculous research and commitment practices as per guidelines of industry, and giving reaal use case studies, we aim towards development of easily deployable and usable world wide spam detection plans based on required use cases. Let it be increasing the accuracy of certain algorithms or getting user feedback on improving the spam detection plans or making spam protective rules and guidelines, the advice given here aims to provide knowledge and can be referenced

to make further advancements in this ever growing passion for providing safety.

1.2 Plan

To make the above requirements possible we developed a feedforward neural network which could easily detect spam messages in social media apps, email etc. The process contains several levels. At the beginning we carefully collected a dataset with a variety of different types of spams. This enabled us to get a feel of how these messages are actually crafted and we can notice certain patterns in case of spam and ham messages. Next we performed lots of pre-processing comprising of cleaning the content i.e reducing unwanted gaps and reducing unwanted errors that might be due to dirty data, We used a process called tokenization and converted these tokens into numbers using numerical conversion which is essential for providing the model as input because neural networks only understand numbers. We researched through articles and read through many papers to find how all these processes work and what role(depth, width, activation functions, and regularization techniques.) each provides in detecting spam efficiently. We were able to build an architecture that could not only use efficient neurons but also accurately predict spam messages. After training the model we use the evaluation metrics that are essential such as accuracy, precision, recall, and F1-Score, to top all this we also did more extra analysis such as ROC curves and Confusion matrix which is a fusion of all the above mentioned evaluation metrics for further information. We were able to perform tuning of parameters and optimize accordingly after each iteration based on the evaluation results which we got after each iteration also including some modifications to model architecture and hyperparameter changes. We also adopted ensemble learning and model distillation methods for efficient and memory reduced models. At the end the model which is trained is sent out as a functional model which could be used as an application or extension towards several use cases to maintain security and credibility in data communication channels. Detailed Report writing has been established which covers all the potential features and implemented features which can be instrumental in giving credibility for comments..

1.3 Related Works

Spam usage methods used before has also used lots of techniques and mechanisms which their goal is to fight a battle against evergrowing spam messages in social media. In previous times the algorithms depended on rule-based [1] systems, filtering of keywords, and blocking or alerting everyone whether sms incoming is spam or not. Although useful for some time, these algorithms don't stand a chance on newer and ever evolving spam generation messages and don't have the capability of finding patterns in the text messages.

In Today's time Artificial intelligence has come to significance which has introduced us to a lot of tools in the internet that can make text generation feel if its written by an actual person, So do we as the security providers step in to detect such anomalies by identifying patterns which feel humanly generated but not actually. Supervised learning, which is learning from someone has many algorithms in this area such as SVMs, decision trees and Naive Bayes [2] classifiers which have gained popularity in machine learning world have been widely explored and gone through for this project. These algorithms have the capability to successfully and effectively predict in large amounts of data where we can provide a conclusion on classification problem by knowing the text content.

Also usage of ensemble methods where we use a variety of algorithms such as random forests and gradient boosting have been utilized to improve performance. Ensemble methods use a variety of separate classifiers to improve model performance by enabling the usage of multiple base classifiers. These methods leverage the variety in separate classifiers to improve generalizing the context and not go towards overfitting, therefore enhancing the effectiveness of spam detection systems.

Deep learning has become so prominent that it has become a universal fitter of almost any data that its given with higher accuracy and precision and can be used for spam detection over traditional machine learning algorithms.Convolutional neural networks (CNNs) [3] which use convolutional layers and recurrent neural networks (RNNs) which use previously used layers to predict the upcoming outcome have been extensively used to this task, which is used to show fantastic results in getting hard patterns and complexities within the text content which is harder to find in machine learning algorithms. These algorithms use large-scale labeled datasets to decide some features directly from raw word embeddings, therefore calling for the need of human made feature engineering

Furthermore, knowledge transfer methods such as already tuned parameters like BERT [4] (Bidirectional Encoder Representations from Transformers), have resulted in coordinating and providing wisdom in neural network architectures to such tasks like spam detection. By using this knowledge learnt from these bags of words we can make some decent predictions which is only possible using this method known as transfer learning for a specifically limited number of labeled data.

In an overview, previous works of spam detection were primarily focusing on rule-based techniques which then went

on to machine learning algorithms which detected patterns to a certain extent then came the neural networks that could accurately fit any data in the universe(almost). While all these approaches have their pros and cons, there has been extensive research in this field to get back with a better detective that catches these criminals.

2. Methodology

2.1 Data Collection

This Method involved various levels of research in collecting datasets over the internet where our requirement was to collect a diverse amount of spam messages where the patterns are variant and not easily catchable by a human, we also wanted a distributed uniformly dataset with more equal number of spam and hams so that it doesn't affect the accuracy of the dataset. All of this was done for careful consideration of bias and variance in the dataset. Variety of publicly available datasets, such as the Enron Email Dataset, SpamAssassin Public Corpus, and UCI Machine Learning Repository datasets, were included in this study. Further emphasis was given to add characteristically different styles of writing and variety of languages, sarcasm and many more types of spam(e.g., phishing emails, promotional emails, etc.) , to improve the model's easiness and context adaptation capabilities.

2.2 Data PreProcessing

Preprocessing is another process which is crucial for the model which is basically if we feed the model poor data then it affects the model accuracy therefore this process is essential. Although we took lots of efforts to get a model that is balanced we got a slightly imbalanced dataset therefore we had to take some measures to make the model balanced. Also we leveraged the tensorflow's tokenizer class which is set with specific tones and an out-of-vocabulary (OOV) token to handle words that are not seen before. This tokenizer is used on the data being trained to find a mapping from words to plain numbers. Furthermore, the context undergoes preprocessing where labels are converted into labelEncoders and the text itself is in vector format using TfidfVectorizer from a module called Scikit learn. This forms a matrix of TF-IDF [5] features, which only takes up 1000 words while excluding the english words which are in the end. This method reduces dimensionality of dataset as well as cites crucial information making the dataset for efficient training with a variety of machine learning models. At the end, the textual data is encrypted into numbers using certain methods and word embeddings, enabling neural networks to learn and use this information efficiently.

Figure 1: Ham Word Cloud

Figure 2: Spam Word Cloud

2.3 Choosing the Model

We selected a feedforward neural network [6] for its robustness and effectiveness over an array of numerical data which we got from word embedding these not only capture complex relationships over large number of words but also is small in terms of usage of convolutional neural networks that usually overfit these kinds of data. The architecture was made in such a way that it could hold up all the complex relationships which we could do over thorough research and design techniques which we saw over the internet. We also considered computational power into considerations as this process has to be easier to do so that it runs in smallest of the smallest systems. We also looked upon how many hidden layers have to be attached by careful consideration and design architecture looked over the internet. We also experimented over various design rules and established this model architecture which finally gave an accurate prediction of spam or a ham classification.

2.4 Training Procedure

In this process we trained the model for 30 epochs with special features such as stopping in the middle and checking after some epochs to get better results by saving the best model in the epochs considered and also saving until there is some high decrease in model accuracy. We stopped early with a threshold of 3 epochs and we evaluated the training procedure where it should stop if no significant increase in model accuracy is shown in three continuous epochs.This method is important as it helps in reducing the chances of overfitting. We used a method to stop the model from training in the middle named as model checkpointing to save the model which is the best in terms of accuracy., we implemented this using the ModelCheckpoint callback a function that callbacks after we get the highest accuracy. We

determined the lowest validation loss,it is saved as filename 'model-{epoch:03d}.keras'. This helped us to save the best model configuration, even if the model's performance degrades in later epochs. We trained the model by executing it with the model.fit() method, where Training_pad and train_labels were provided as the training dataset, and Testing_pad and test_labels were provided as the validation dataset. We conducted the training in a verbose mode set to level 2, which provides a concise, epoch-by-epoch update, which helped us to closely monitor the model's learning progress. This approach not only minimizes the loss on the training set but also improves the model's usefulness in generalization of new, unseen data efficiently.

2.5 Hyperparameter Tuning

In this project, we performed hyperparameter tuning to optimize the neural network model's performance. We focused on parameters which had a great impact on the model performance like the depth of the vector. we had set the number of epochs to 30, we came up with this number after a lot of testing with different values to prevent overfitting and to make the model learn only the important features. We also used an early stopping feature, which played an important role to prevent overfitting. We had set the threshold of early stopping (patience value) as 3, which means training would be stopped automatically if there was no improvement in loss which will be monitored for three continuous iterations. By using this strategy it helped us to prevent overfitting and made us sure that the model works well to test data by stopping the training at the certain moment. We also used model checkpointing to save the best model state at each epoch where we found an improvement in test loss. We implemented this using the save_best_only=True setting. Using a setup made us confident that the best-performing model is retained, so that only the best model will be chosen for implementation. Thus we used hyperparameter training so that we could give our best to make the model perfect.

2.6 Model Evaluation

The efficacy of the trained feedforward neural network in spam recognition was assessed by extensive testing on test data. Recall, accuracy, precision, and F1-score were among the performance metrics that were computed to assess the model's capacity to differentiate between spam and ham letters.Evaluation of the trained model using multiple metrics, including F1 score and recall, revealed excellent performance. This illustrates the degree of accuracy in spam detection.Roughly 91.5% of the events in the model were properly detected overall. Precision, which is used to measure the accuracy of positive prediction, was 92.73%. This made us clear that when our model predicts an instance as positive, it will correct about 92.73% of the time. Another matrix that we used was recall score, which shows how well our model will find all the important things in the text data that we had provided.The score that we got was 90.74%, which showed us that our model was able to capture correct predictions. We also used the F1 score matrix, which is calculated based on precision and recall. The score that we had got was 91.62%, showing that our model was performing well on all the matrices which we had used. We also made a confusion matrix which provided more insight. The results that we had got were that the model correctly identified 1059 true negatives(TN) and 2731 true positives(TP),false negatives (154) and false positives (126). This matrix helped us to make sure that our model was performing well and was able to give better results.

```
76/76 [==============================] - 0s 3ms/step
Accuracy: 0.9155925155925156
Precision: 0.9273182957393483
Recall: 0.9053833605220228
F1 Score: 0.9162195625257944
[[1092   87]
 [ 116 1110]]
```

Figure 3: Evaluation Matrix

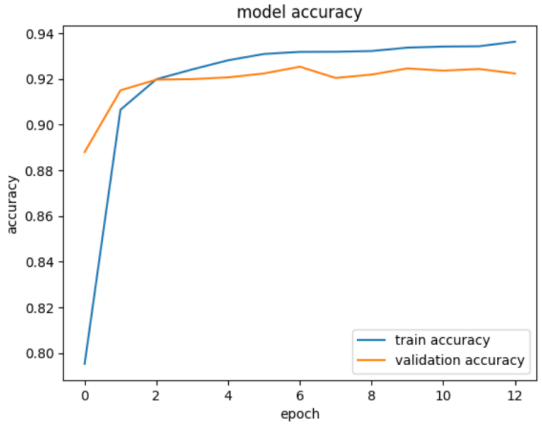

Figure 4: Model Accuracy

3. Model Architecture

We came up with the design of the model which is a feedforward sequential neural network [6] where we use certain design principles which govern neural networks with extensive hyperparameter tuning as well as optimization to fit the needs of a spam detection system which requires small memory capacity for smaller systems with higher levels of security added into it. The model primarily consists of an embedding layer which is continued by a global average pooling layer and ending with a dense layer, dropout layer and final output layer.

We give a comprehensive role of each layer here in this sequence:

Embedding Layer: Primary layer where the shape of the

output is (None, 50, 16). None denotes the size of the batch,which can depend on how many dataset samples are processed by the neural network at a given amount of time. The layer according to the design principles is used to take 50 tokens per sentence and conversion is done into a 32-dimensional vector. This layer helps in converting sparse tokens which are in category into a more concise vector space which is interrelatable. This makes it easier for the model to handle text data effectively. It has 8,000 parameters, this makes the creation of varieties of vocabulary that helps in learning a comprehensive bunch of word representations, which tells us that the size of the vocabulary might be around 500 (8000/16). The layer is important for small ranged NLP tasks which takes number indices into dense vector format and engages in semantic representation of words.

Global Average Pooling 1D (GlobalAveragePooling1D): After the embedding layer we had put a global average pooling layer. We had put this layer because the pooling layer helps us to reduce the output dimension size so that we can train the model faster but we are also preventing the information the earlier layer had learnt. We were able to reduce the feature map size to 50 which will help us to make our training process faster. By using this layer we reduced the number of parameters, which will help us save training time. After this layer each sample was only a vector of depth of 16 this helped us to make it easier to connect to dense layers. This layer doesn't have any parameters because it only averages the values.

Dense Layer: After the global average pooling layer we had put a dense layer. We had configured this layer to have 32 neurons. This layer that we used was a fully connected layer which means that each input is connected to every node or neuron. Since we had configured this layer to have 32 neuron, the total parameter that we had for this layer was 544 parameters (16 from the previous layer, like that there are 32 units plus 32 bias terms).This layer will help us to capture the important features which all the previous layers we able to understand

Dropout Layer: After the dense layer we had put a dropout layer. This layer has no parameters as this player is not performing any calculation. We had put this layer because we wanted to control overfitting. What this layer does is that it randomly makes some data 0 so that it becomes harder for the model to predict stuff. This will help us to prevent overfitting. To this layer we have to put a threshold value that determines how many nodes should be turned off. We had put a value of 0.3 that means 30% of the output will be turned off.

Output Dense Layer: After the dropout layer we had put our final layer which is another dense layer. But we had configured this layer to have only 1 neuron. This neuron's job is to give a prediction score based on the features that the previous layers have learned that determines if a text is spam or not. We had configured this layer with one neuron therefore this layer has 65 parameters out of which 64 are the data from the previous layers and 1 is the bias .

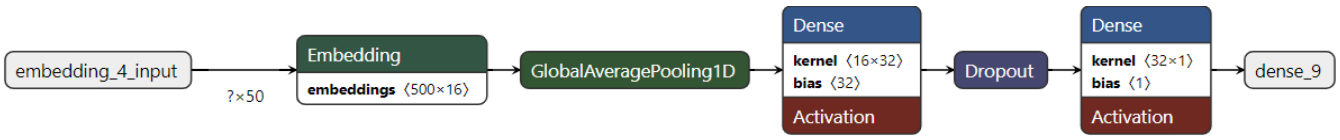

Figure 5: Feed Forward Neural Network Architecture

Figure 6: FFNN with Convolution Layer

Overall, the model that we have built after adding all the parameters has 8,577. We are confident that the model that we have built is very powerful to classify whether a text is spam or not. Since we have used many deep learning layers like embedding, dense layers we can say that our model will give the best output.

We had also implemented another model that used a convolution layer. But we had certain problems with that model.

1D Convolution Layer (conv1d_2): We had put a convolution layer [7] after the embedding layer, we had configured this layer to have 32 filters which we thought would be helpful to analyze and detect features in much more detail. The reason we had put this layer was to make our model learn more features in depth. The output from this layer was (None, 46, 32) .

Batch Normalization (batch_normalization_2): We had added a Batch Normalization layer after the first dense layer in our model. We had an idea to normalize the data coming out of the dense layer as it would help us to train much faster.

But there was a problem with this model architecture. We initially thought this model would be better but we were wrong. This model became overfit on the training data so it was hard for it to perform on testing data giving us less accuracy. But this model had performed better than all the traditional Ml algorithms which we had tested. So we rejected this model and used our own simple feed forward neural network.

4. Experimental Results

4.1 Traditional ML Model Results

There might be a question that arises in why we take feedforward neural networks over traditional machine learning algorithms [8]. The answer is that we find results which can not be found in traditional machine learning algorithms where we found that the neural network could effectively capture some complex patterns and relationships which cant be found using traditional algorithms. Also the effectiveness of neural networks in fitting all possible data over a certain data just by some effective hyperparameter tuning and optimization to suit our needs of spam detection makes it a better fit. Among various Machine learning models which are evaluated—Random Forest, Support Vector Machine (SVM), Naïve Bayes (GaussianNB), and Decision Tree—Random Forest was the best among them all, defeating its competitors with its highest accuracy of 88.32%. This model leveraged ensemble methods that handle the variability in spam email characteristics, thereby providing a mechanism against overfitting while having high accuracy. SVM also showed good performance as it is mostly preferred in classification techniques, with accuracy 86.61% , indicating its usability in searching and finding high-dimensional spaces which are present in text data.

Other models such as the Naïve Bayes and Decision Tree [9] models have shown significantly lower accuracy with Naïve Bayes achieving around 81% accuracy and Decision Tree model with 81.33%. Yet Naive Bayes is simple and is effective for computationally easier usage purposes, its thinking on feature independence is flawed, especially in case of complexities in spam detection procedures where this feature independence is rare to see. Decision trees are visually easy to recognize but miss crucial features unless we adjust them with a lot of optimization which is not suitable in general spam detection techniques. They can either be too simple or too complex which can be an overhead in both cases.

4.2 Feed Forward Neural network Results

Neural networks outclassed all these traditional machine learning algorithms with an impressive accuracy of 93%. This high performance is consistent in all evaluation metrics which certify a model such as precision, recall, and F1-score. The precision of the model was 91.17 and recall of 92.74 and the F1 score was 91.95. This ability comes from a crucial technique in neural networks which is backpropagation which knows what to correct and goes on reducing the loss to suit its needs and predict accurately a result which has to be established. This ability helps in spotting spam patterns of words which are being reported continuously. Neural networks can adjust easily to continuously changing spam generation techniques.

With every new spam generation technique coming day by day there is a neural network that can train itself to capture such results, the design of such feedforward neural networks is necessary to avoid spam attacks where the link would crash the laptop or could prompt the user to pay a little amount of money and get back 10 times the amount and so on. Therefore this is a big advantage of adding security into the systems which we are using on a daily basis.

In summary, traditional machine learning algorithms like random forest and SVM [10] provide many options in spam detection. The overall performance of neural networks is better overall with better ability to consider spam tactics. This makes it the most suitable option to be used in real time spam detection and evolution in such design will make it more and more progressive in keeping up with spam attacks. This thorough report and study shows us how important it is to pick a model and the importance of design principles in model performance with a pinch of experimentation to secure digital media from spam attacks.

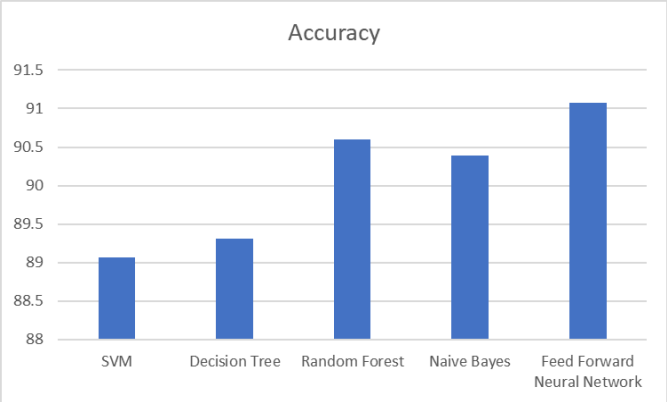

Figure 7: Comparision of Accuracy

```
SVM Confusion Matrix:
 [[2775  138]
 [ 185  972]]
Decision Tree Confusion Matrix:
 [[2679  234]
 [ 207  950]]
Random Forest Confusion Matrix:
 [[2787  126]
 [ 168  989]]
Naive Bayes Confusion Matrix:
 [[2747  166]
 [ 225  932]]
```

Figure 8: Evaluation Matrix of Traditional ML

5. Conclusion/ Future Work

Overall we were able to prove how design principles can affect model performance and how feedforward neural networks [6] emerged as a clear winner against traditional machine learning algorithms. We did this by comparing the model performances of each model where we also studied about the depths of the evaluation metrics of each model to come to a conclusion.

Our model has an exclusive architecture wherein we have an embedding layer, global average pooling, dense layers with improved and enhanced amount of neurons, and dropout regularization, has won the battle against spam detection from real ham messages. The achieved 91.36% accuracy indicates the model's ability to classify emails and SMS as spam or not spam.

Furthermore we plan to deploy this model as a chrome extension wherein we will be able to get the context of the message using native javascript if the site authorizes or we would use API's to get the data where will put that to the model using hugging face. We will deploy our model in hugging face and use it as a link to accurately predict if the recent message was a ham or a spam which will reduce potential risks in social media sites.

For the future of this project we are thinking of further fine-tuning or simplifying the model to get better results. We will explore the world of LSTM [11] and 1D convolution.We will try to further tune the hyperparameter so that we achieve even better results. We will also look forward to making our own dataset which will be more balanced text on spam and ham.

In conclusion the project indicates a new beginning for many more research and implementations in this field with recognized design methodologies for the effectiveness and security of the digital world that we live in.

6. References

[1] D. V. R. D. J. S. S. Abiramasundari, "Spam filtering using Semantic and Rule Based model via supervised learning," *Annals of RSCB,* p. 3975–3992, 2021.

[2] W. a. S. J. a. Z. L. a. C. C. a. Y. Q. Feng, A support vector machine based naive Bayes algorithm for spam filtering, 2016.

[3] T. Huang, A CNN Model for SMS Spam Detection, 2019.

[4] G. N. a. M. {. K. a. M. Y. a. B. Z. a. M. {. Hanif}, "Email spam detection by deep learning models using novel feature selection technique and BERT," *Egyptian Informatics Journal,* vol. 26, p. 100473, 2024.

[5] A. Aizawa, "An information-theoretic perspective of tf–idf measures," *Information Processing & Management,* vol. 39, pp. 45-65, 2003.

[6] M. H. Sazlı, "M. H. Sazlı, "A brief review of feed-forward neural networks," *Commun.Fac.Sci.Univ.Ank.Series,* 2006.

[7] "1D convolutional neural networks and applications: A survey," *Mechanical Systems and Signal Processing,* vol. 151, p. 107398, 2021.

[8] M. K. T. P. J. Crawford, "Survey of review spam detection using machine learning techniques," *Journal of Big Data 2,* 2015.

[9] M. C. M. McCord, "Spam Detection on Twitter Using Traditional Classifiers," *Lecture Notes in Computer Science,* vol. 6906, 2011.

[10] R. C. a. P. D. R. Patil, "IEEE 9th International Conference on Intelligent Systems and Control (ISCO)," *Web spam detection using SVM classifier,* pp. 1-4, 2015.

[11] G. S. M. &. A. B. Jain, "Optimizing semantic LSTM for spam detection," 2019.

[12] S. Suthaharan, Machine Learning Models and Algorithms for Big Data Classification: Thinking with Examples for Effective Learning, Springer US, 2015.