



SPAM FILTERING USING BIG DATA AND DEEP LEARNING

ONUR GÖKER

FEBRUARY 2018

SPAM FILTERING USING BIG DATA AND DEEP LEARNING

**A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED
SCIENCES OF
ÇANKAYA UNIVERSITY**

**BY
ONUR GÖKER**

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF
COMPUTER ENGINEERING**

FEBRUARY 2018

Title of the Thesis: **Spam filtering using big data and deep learning**

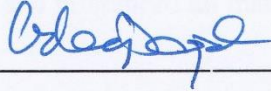
Submitted by **Onur GÖKER**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University.


Prof. Dr. Can ÇOĞUN


Director

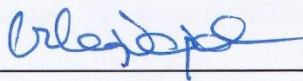
I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.


Prof. Dr. Erdoğan DOĞDU

Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.


Assist. Prof. Dr. Roya CHOUPANI
Co-Supervisor

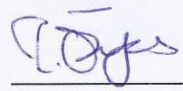

Prof. Dr. Erdoğan DOĞDU
Supervisor

Examination Date: 09.02.2018

Examining Committee Members

Assoc. Prof. Tansel Özyer

(TOBB ETU)



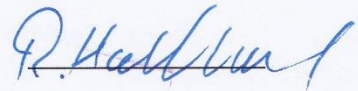
Prof. Dr. Erdoğan Dođdu

(Çankaya Univ.)



Assoc. Prof. Reza Zare Hassanpour

(Çankaya Univ.)



STATEMENT OF NON-PLAGIARISM PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Onur GÖKER

Signature :



Date :

09.03.2018

ABSTRACT

Spam Filtering Using Big Data and Deep Learning

GÖKER, Onur

M.Sc., Department of Computer Engineering

Supervisor: Prof. Dr. Erdoğan DOĞDU

Co-Supervisor: Assist. Prof. Dr. Roya CHOUPANI

February 2018, 70 pages

Spam e-mails and other fake, falsified e-mails like phishing are considered as spam e-mails, which aim to collect sensitive personal information about the users via network or behave against authority in an illegal way. Most of the e-mails around the Internet contain spam context or other relevant spam like context such as phishing e-mails. Since the main purpose of this behavior is to harm Internet users financially or benefit from the community maliciously, it is vital to detect these spam e-mails immediately to prevent unauthorized access to email users' credentials. To detect spam e-mails, using successful machine learning and classification methods are therefore important for timely processing of emails. Considering the billions of e-mails on the internet, automatic classification of emails as spam or not spam is an important problem. In this thesis, we studied supervised machine learning and specifically "deep learning" methods to classify emails. Our results indicate that deep learning is very promising in terms of successful classification of emails with an accuracy of up to 96%.

Keywords: Spam filtering, spam detection, email classification, classification, supervised learning, deep learning, cybersecurity

ÖZ

Büyük Veri ve Derin Öğrenmeyi Kullanarak Spam Filtreleme

GÖKER, Onur

Yüksek Lisans, Bilgisayar Mühendisliği Anabilim Dalı

Tez Yöneticisi: Prof. Dr Erdoğan DOĞDU

Eş - Tez Yöneticisi: Yard. Doç. Dr. Roya CHOUPANI

Şubat 2018, 70 sayfa

İstenmeyen (spam) e-postalar veya diğer oltalama (phishing) gibi sahte e-postalar, küresel ağ aracılığıyla hassas kişisel bilgi toplamayı amaçlayan veya illegal işlem yapmaya yönelik zararlı e-postalar olarak düşünülür. İnternette dolaşan birçok e-postanın içinde istenmeyen içerik bulunur ya da bu tür aldatici e-postalar oltalama gibi diğer sahte e-postalara benzer. Bu davranışın asıl amacı kullanıcıya fiilen zarar vermek veya toplumdan haksız çıkar sağlamak olduğundan, bu istenmeyen e-postalar aracılığıyla yapılan, kullanıcıların / müşterilerin kimlik bilgilerine yetkisiz erişimin önlenmesini derhal tespit etmek ve bu tespit için başarılı sınıflandırma yöntemleri kullanmak önemli rol oynamaktadır. İnternetteki milyarlarca e-postayı göz önünde bulundurursak, e-postaların temiz ya da sahte olup olmadığının otomatik olarak sınıflandırılması önemli bir sorundur. Bu tezde, e-postaların sahte olup olmadığıyla ilgili sınıflandırma yapmak için denetimli makine öğrenmesi ve özel olarak derin öğrenme metotları kullandık. Sonuçlarımızın da belirttiği gibi, derin öğrenmenin e-posta sınıflandırması yapmada %96 başarı oranıyla kayda değer bir etkisi vardır.

Anahtar Kelimeler: Spam filtreleme, spam algılama, eposta sınıflandırma, sınıflandırma, denetimli makina öğrenmesi, derin öğrenme, sibergüvenlik.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Prof. Dr. Erdoğan DOĞDU for his supervision, special guidance, suggestions, and encouragement for my thesis work. Additionally, thanks for the support of Assist. Prof. Roya CHOUPANI during the study. I would also like to thank Assoc. Prof. Reza Zare Hassanpour for his valuable suggestions and comments for this work.

I dedicate this thesis to my family, who are in important part in my life. Without their support, encouragement, and love, none of this would be possible.

I would like to thank my company, Comodo Inc. for their giving me time and support during my MS studies. Especially, I would like to thank my colleagues and mentors Nurettin Mert Aydın, Development Manager in ASLab Project and Hatice Sakarya, an expert in the R&D Investments Office at Comodo.

I want to thank to my friend Nazlı Nazlı for her cooperation, help, and support during my thesis studies. I also want to thank my friends Gökhan Tamkoç, Ali Abbasi, and Negin Bagherzade for their help and support.

TABLE OF CONTENTS

STATEMENT OF NON-PLAGIARISM PAGE	iii
ABSTRACT	iv
ÖZ	v
ACKNOWLEDGEMENTS	vi
TABLE OF CONTENTS	vii
LIST OF TABLES	x
LIST OF FIGURES	xi
LIST OF ABBREVIATIONS	xiii
CHAPTER 1	1
INTRODUCTION	1
1.1 Background	1
1.2 Problem Statement	2
1.3 Contributions.....	5
1.4 Thesis Organization	6
CHAPTER 2	7
RELATED WORK	7

2.1 Rule Based Detection	8
2.2 Machine Learning Based Spam Detection	9
2.3 Deep Learning-based Spam Detection	10
CHAPTER 3	12
DEEP LEARNING-BASED SPAM CLASSIFICATION	12
3.1 Data Representation	12
3.1.1 Weighted TF-IDF Vectorization	13
3.1.2 TF-IDF using SciKit Learn	14
3.1.3 Word2Vec	14
3.2 Machine Learning Based Classification	15
3.3 Deep Learning-Based Classification	16
3.3.1 Multilayer Perceptron	16
3.3.2 Logistic Regression	17
3.3.3 Keras on TensorFlow	17
CHAPTER 4	18
EVALUATION	18
4.1 Datasets	18
4.2 Tools and Libraries	19
4.3 Evaluation Metrics	20
4.3.1 Test Plans	21
4.3.2 Results	22
4.3.2.1 Results with Weighted TF-IDF Vector Representation	22

4.3.2.3 Results with SciKit Learn TF-IDF Vector Representation	33
4.3.2.4 Results with Word2Vec Vector Representation	34
CHAPTER 5	46
CONCLUSION.....	46
REFERENCES.....	47
CURRICULUM VITAE.....	53



LIST OF TABLES

1. Table 1- Spam Detection Literature Taxonomy	8
2. Table 2- Vector names, generation methods and their sizes.....	19
3. Table 3- Relationship between scores on results	21
4. Table 4- Results for Weighted TF-IDF method on WEKA.....	23
5. Table 5 – F Measure Results for 300 + 300 and 500 + 500 ham – spam datasets	27
6. Table 6– F Measure Results for 1000 + 1000 and 2000 + 2000 ham – spam datasets	28
7. Table 7– F Measure Results for 5000 + 5000 and 10000 + 10000 ham – spam datasets	29
8. Table 8– 10-Fold Cross Validation Results in TensorFlow + Keras	33
9. Table 9- Test Results for SciKit Learn	34
10. Table 10– Weka Results for Word2Vec implementation	35
11. Table 11- F Measure Results for Word2Vec 300 + 300 and 500 + 500 implementation.....	39
12. Table 12- F Measure Results for Word2Vec 1000 + 1000 and 2000 + 2000 implementation.....	40
13. Table 13- F Measure Results for Word2Vec 5000 + 5000 and 10000 + 10000 implementation.....	41
14. Table 14-TensorFlow Results for all data sets with Word2Vec	45

LIST OF FIGURES

1. Figure 1- Phishing E-mail Process [1]	2
2. Figure 2- Sample Neural Network including Hidden Layers [14].....	5
3. Figure 3- Vectorization of words in Word2Vec [39].....	15
4. Figure 4- Multilayer Perceptron with 1-hidden layer	16
5. Figure 5- Accuracy Comparison of Weighted TF-IDF method algorithms for 300 ham + 300 spam datasets.	24
6. Figure 6- Accuracy Comparison of Weighted TF-IDF method algorithms for 500 ham + 500 spam datasets.	24
7. Figure 7- Accuracy Comparison of Weighted TF-IDF method algorithms for 1000 ham + 1000 spam dataset.	25
8. Figure 8- Accuracy Comparison of Weighted TF-IDF method algorithms for 2000 ham + 2000 spam dataset.	25
9. Figure 9- Accuracy Comparison of Weighted TF-IDF method algorithms for 5000 ham + 5000 spam datasets.	26
10. Figure 10- Accuracy Comparison of Weighted TF-IDF method algorithms for 10000 ham + 10000 spam datasets.	26
11. Figure 11- Algorithm Comparison for 300 + 300 Data Set	30
12. Figure 12- Algorithm Comparison for 300 + 300 Data Set	30
13. Figure 13 - Algorithm Comparison for 1000 + 1000 Data Set	31
14. Figure 14- Algorithm Comparison for 2000 + 2000 Data Set	31
15. Figure 15- Algorithm Comparison for 5000 + 5000 Data Set	32
16. Figure 16- Algorithm Comparison for 10000 + 10000 Data Set	32
17. Figure 17- TensorFlow with Keras for all data sets	33

18. Figure 18- Accuracy Results for SciKit Learn.....	34
19. Figure 19 - Success Ratios for Word2Vec 300 + 300 data sets.....	36
20. Figure 20- Success Ratios for Word2Vec 500 + 500 data sets.....	36
21. Figure 21- Success Ratios for Word2Vec 1000 + 1000 data sets.....	37
22. Figure 22- Success Ratios for Word2Vec 2000 + 2000 data sets.....	37
23. Figure 23- Success Ratios for Word2Vec 5000 + 5000 data sets.....	38
24. Figure 24- F Measure Graphics for Word2Vec 300 + 300 data sets.....	42
25. Figure 25 - F Measure Graphics for Word2Vec 500 + 500 data sets.....	42
26. Figure 26- F Measure Graphics for Word2Vec 1000 + 1000 data sets.....	43
27. Figure 27- F Measure Graphics for Word2Vec 2000 + 2000 data sets.....	43
28. Figure 28- F Measure Graphics for Word2Vec 5000 + 5000 data sets.....	44
29. Figure 29- F Measure Graphics for Word2Vec 10000 + 10000 data sets.....	44
30. Figure 30- Accuracy Results for Tensorflow with Word2Vec.....	45

LIST OF ABBREVIATIONS

ACM	Association for Computing Machinery
APWG	Anti-Phishing Working Group
BLEU	Bilingual Evaluation Understudy
CASSANDRA	Collaborative Anti-Spam System Allowing Node- Decentralized Research Algorithms
CNN	Convolutional Neural Network
DBN	Deep Belief Network
DL	Deep Learning
FN	False Negative
FP	False Positive
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
IDE	Integrated Development Environment
IDF	Inverse Domain Frequency
IEEE	The Institute of Electrical and Electronics Engineers

IP	Internet Protocol
LSTM	Long-Short Term Memory
MAAWG	Messaging Malware Mobile Anti-Abuse Working Group
MNIST	The Mixed National Institute of Standards and Technology
ML	Machine Learning
MLP	Multi-Layer Perceptron
MAPS	Mail Abuse Prevention System
NB	Naive Bayes
NLTK	Natural Language Toolkit
NN	Neural Network
RNN	Recurrent Neural Network
SVM	Support Vector Machine
TF	Term Frequency
TN	True Negative
TP	True Positive
WWW	World Wide Web

CHAPTER 1

INTRODUCTION

1.1 Background

Technological developments made our lives easier and brought us convenience. When considering the Internet of Things, remote management, digital contents (a.k.a. Multimedia content), online shopping, social media platforms like Facebook, Twitter, Instagram, cloud systems like Google Drive, Microsoft's OneDrive, Dropbox and messaging systems like WhatsApp, Viber, WeChat, it is a great convenience to include technology in our daily lives. Besides the advantages, technology also made it easier for the criminals to do their acts online, such as stealing credit card numbers or sensitive information from users, like e-mail, user ID and/or passwords, hacking social media accounts, taking control of devices or integrated systems. With the evolution of technology, cybersecurity became a very essential problem for everyone. Spam e-mails, including phishing e-mails, are also part of the cybersecurity issues as well. Spam e-mails are considered a potential problem all around the world. Spam e-mails especially target people, who are involved in financial transactions over the internet. Spam e-mails, and phishing e-mails as well, intend to grab customers' user credentials, credit card numbers and more. Clearly, the main purpose of this behavior can be defined as damaging users financially or resist against public authority in illegal ways. Figure 1 illustrates phishing process in real world as pointed out in [1]:



Figure 1- Phishing E-mail Process [1]

Messaging Malware Mobile Anti-Abuse Working Group (MAAWG) (an industry association against botnets, malware, spam, viruses, DoS attacks and other online exploitation) presented a report about spam e-mails. According to the MAAWG's report, almost 75%-80% of total e-mails could be counted as spam e-mails [2]. According to Spam Filter Review results, total spam mail count was close to 41 billion per day, which is equal to 40% of the total e-mail in 2003 [3]. And, according to the same group's latest report¹ in 2014, 90% of all email is spam. We understand from these numbers that spam e-mail is becoming a major problem every day. Therefore, it is vital to detect and stop spreading spam e-mails automatically.

1.2 Problem Statement

With the distribution of broadband and mobile internet around the world, being online on the network became easier and cheaper. Internet's wide spread usage has begun with e-mails first and then later with the invention of World Wide Web (WWW). In 1989, Tim Berners-Lee, a scientist at CERN, invented this technology and named as "World Wide Web" [4]. People all around the world started to take part in this new world by first using dial-up connections either 28K or 56K speeds. When broadband and satellite connections took the charge, internet users became more active on the internet. They started watching videos, commenting on websites, shopping online, engaging on social networks and other online processes. With the usage of these broadband internet culture, Web 2.0 took an important role. Unfortunately, internet is also a place for criminals and abusers.

¹ <https://www.m3aawg.org/for-the-industry/email-metrics-report>

When considering this kind of complex, global and huge network, it is not easy to control or even to monitor it. Therefore, internet security became a major problem for everyone. This is now called "cyber security".

E-mail is a major concern in terms of security. E-mail might be considered as just a communication tool. But, when dealing with other tools like adding attachments to emails, using HTML characters in the text, an e-mail is also a potential threat at the same time. Even when considering the governmental issues, politics and social movements, spread of unwanted e-mail may cause severe problems in the world as well [5].

A spam e-mail could best be defined as unsolicited email that include unwelcome content to benefit financially or cause harm or annoyance to internet users [6]. For example, it is very common to the following in everyday email traffic for all email users:

- Multimedia like image, sound, video containing viruses
- Attachments like zip or executable bat files containing malwares
- Links used to redirect users to make phishing
- All sorts of advertisements

It might be easy to detect an e-mail whether it is potentially dangerous or useful with bare eyes, but when working with billions of emails at the same time, it is not easy to analyze e-mails automatically and urgently. Since performance and security take important roles as non-functional requirements in internet systems, an automated system is needed to make the analysis and classification.

To automate the spam and ham classification, analyzing the content is required. All email content includes the following [7]:

1. Sender
2. Receiver
3. Title
4. Body
5. Header Information

Since the main research problem in this thesis is to determine whether an e-mail is potentially dangerous (spam) or clean (ham), and it is also important to do this timely with high performance, we limit the email content analysis to email's title and body text only.

To classify an email as clean (ham) or spam in a limited time, we need to consider an automated detection system. And machine learning allows us to make automatic classification in a limited time by using techniques such as text categorization. Making text categorization with Machine Learning has started to become popular in the 1990s [8].

Machine Learning has also been used in the other areas in 2000s. In [9], the authors have for example researched speech categorization. In [10], authors researched about image categorization using machine learning.

When applying Machine Learning, defining the methodology takes an important role when applying an algorithm. These methodologies are defined as follow:

- Supervised
- Unsupervised
- Semi-supervised

These methodologies are categorized regarding to known output Y of a given X. In case classification is made regarding to known output Y, methodology is referred as Supervised (a.k.a. labeled). In case of unknown output, methodology is counted as unsupervised.

In case of some known output Y, methodology could also be counted as Semi-supervised as well [11]. Following algorithms could be defined as good examples of

Machine Learning algorithms: Naive Bayes, Bayesian, SVM and other algorithms [12].

When considering the Bayesian algorithms on text classification like e-mail body or e-mail title by using bigger datasets, it might be better to use alternative multi-layer algorithms.

Deep learning algorithms use such multi layers for transforming the input raw data to an abstracted form of it. When making a process on semantic texts like words in a sentence, using deep learning tools might perform with better results. To classify millions of e-mails, it is better to use deep learning techniques instead of Bayesian machine learning techniques in the light of the information given in [13]. Figure 2 describes the multiple hidden layers in a sample neural network.

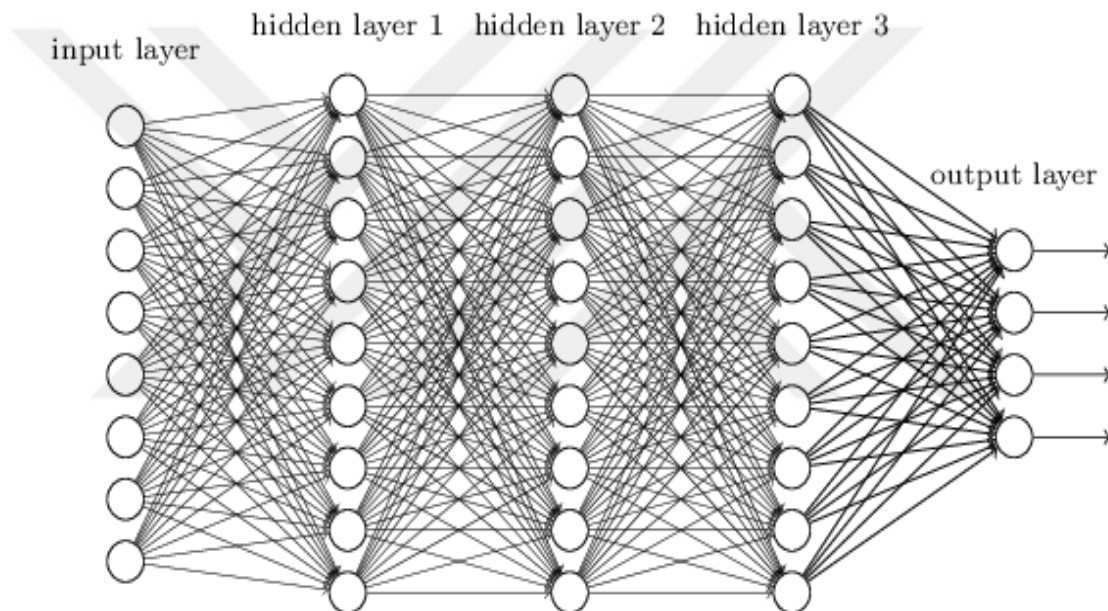


Figure 2- Sample Neural Network including Hidden Layers [14]

1.3 Contributions

Our contributions in this thesis are as follows:

- We preprocessed and sampled Enron email dataset for training and testing machine learning and deep learning algorithms on alternative data representation methods.

- We developed a TF-IDF based vector representation method for email textual data with the purpose of identifying spam vs ham emails.
- We compared several machine learning and deep learning algorithms on sampled email datasets of varying sizes and reported on the performance of different data representation methods towards a better spam detection method.

1.4 Thesis Organization

This thesis is divided into five chapters:

Chapter 1 is the introduction and background part which contains the definition and effects of spam filtering. This part also reveals the foundation of some disciplines that include deep learning, machine learning, neural networks and big data as well.

Chapter 2 is about literature review on email detection and filtering. We also present the taxonomy of the related work in this area and present a thorough analysis.

Chapter 3 explains our methodology for spam filtering and classification. We define our data representation method, based on word frequencies and semantic relationships. Then, deep learning-based classification methods we used are explained.

Chapter 4 presents the results of our experiments. We first present the datasets we used, how we obtained, processed and prepared for classification. Then the results are presented and evaluated in detail.

Chapter 5 is the conclusion part. Here we summarize our work and point to future work in this area.

CHAPTER 2

RELATED WORK

Spamming via e-mail started very early in the 1990s, when the commercial side of Internet is revealed [15]. Although it became popular in 1990s, the first spam e-mail started in 1978 via ARPANET by Gary Thuerk, who was a marketer for the company DEC [16]. After the quick increase of volume of the spam e-mails in just a few years around the world, internet community started to look for a solution and in 1996 Mail Abuse Prevention System [17] is founded by Dave Rand and Paul Vixie to prevent spam e-mails by tracking IP Addresses.

Today, spam detection is considered an important precaution for security in all areas of the Internet. These areas contain especially websites and e-mails servers. Spam detection and filtering in emails have been studied for quite some time. Our literature survey found that automatic spam detection is concentrated in three main methods: rule-based methods, machine learning-based methods, and recently deep learning-based methods, which is a subdomain of machine learning [18]. In this chapter we review these works.

Table 1 summarizes the taxonomy of our findings in the literature. It lists the works we found in three different methodologies along with a summary of algorithms used, datasets tested, and the success rates obtained.

Table 1- Spam Detection Literature Taxonomy

Method	Algorithm	Data Sets	Highest Success Rate	References
Rule Based	Cassandra		99.59%	[63], [64]
Machine Learning	Bayesian Net	CSDMC2010, Spam Assassin, and LingSpam, 1171 raw phishing emails and 1718 legitimate emails	85.45%	[55], [59]
	Naive Bayes	Discretized, RUL:6000 emails with the spam rate 37.04%	99.46%	[27], [30], [48], [54], [55], [57], [61]
	SVM	1171 raw phishing emails and 1718 legitimate emails, Discretized, RUL:6000 emails with the spam rate 37.04%	96.90%	[22], [30], [48], [49], [50], [51], [52], [54], [55], [58], [59], [60], [61]
	J48	4601 messages: 1813 (%39) by Hopkins et al as spam, others are Legal messages by Forman.	92.6 %	[11], [23]
	Random Forest	4601 messages: 1813 (%39) by Hopkins et al as spam, others are Legal messages by Forman.	93.75%	[48], [53], [56]
Deep Learning	MLP	MNIST handwriting data set NORB Dataset	99.04%	[22], [62], [26]
	Logistic Regression	1171 raw phishing emails and 1718 legitimate emails,	88.59%	[19], [59]

2.1 Rule Based Detection

Rule-based detection is considered one of the early methods in spam detection. It is based on very different rules such as IF 'condition' THEN 'result' type of rules, considering the source of email, word usage, etc.

In [19], authors have proposed a rule-based detection method by using disjunctive normal form (DNF) decision rules. By using 10-fold cross-validation on 5,000 training cases and 10,000 cases for independent testing, they achieved almost error rates 0.40 and below.

Wu developed a hybrid method of rule-based techniques and neural networks [48]. Their rule-based method depends on identifying the spamming behaviors observed from the headers and system logs of emails, which is transformed into a digital format. Their method is not based on the use of keywords but the spamming behaviors as features of emails. Then, they use the neural network to classify emails. Gray and Haahr [64] studied collaborative filtering by designing an architecture for such an email system, where email is processed in a centralized server and can be filtered using the users' feedback to the system. This approach assumes that the spam email is sent to many users on the same server. Therefore, it can be considered a crowdsourcing approach and the system only facilitates such human filtering.

2.2 Machine Learning Based Spam Detection

There are many recent works in machine learning-based spam detection. Many different algorithms have been used and tested, including Naïve-Bayes, Bayesian Net, SVM, decision trees (J48), random forests, and so on, with very high accuracy results. Therefore, machine learning-based methods are successful in spam classification. But, these are tested on specific or propriety datasets and therefore with the spam getting more sophisticated there is always a need for more sophisticated solutions in automated spam detection. And, currently machine learning is the only way to do this considering the every increasing email traffic and web data.

Just to give a few detailed examples from these works, in study [20], the authors used 10-fold cross-validation with 23 features and apply Random Forest, J48 and PART algorithms. They obtained 98.87%, 98.11% and 98.10% accuracy rates respectively. In work [21], the authors have tried multiple learning algorithms, as well as different datasets like 1000 spam and ham e-mails of Enron and Ling Spam Dataset with 95% success rate, compared to 83% accuracy with Naive Bayes, 86% with LMT, and 78%

with J48. In [22], the authors have proposed a different method rather than using ordinary Support Vector Machines (SVM) or Naive Bayes algorithms, which was named as “Cumulative Weighted Sum”, to get better success.

2.3 Deep Learning-based Spam Detection

Machine Learning (ML) techniques are being used in every corner of our lives from web searches, to content filtering, recommendations on e-commerce web sites, to self-driving cars, and many other operations. ML systems are identifying objects in images or videos, transcribing speech to text, finding related news items and posts, and they are used in products depending on past user behaviors [23].

Neural networks-based machine learning is also a type of machine learning technique, imitating the neural structure of human brain. Artificial Neural Networks (ANN) enable machines or computers learn from observed data². They are used in pattern recognition, classification, image recognition, and all sorts of machine learning problems from the start in the last half of 20th century. Deep learning is later introduced and applied on neural networks, allowing neural networks to work better multiple hidden layers. With the evolution of technology, higher computational power in computers and big data techniques, deep learning and other neural network-based approaches bring us the opportunity to work with multiple hidden layers on data. Some of the attempts to use deep learning in spam detection are reviewed in the following.

Twitter needed to find a better solution for spam detection than machine learning. Twitter is an expanding social media pioneer, which provides a sharing environment to post 140-character long short texts to present users’ ideas, emotions or even daily moments among Twitter network users [24]. This huge network with millions of users, dealing with big data contains vicious people, who are intended to create malicious content as well [25]. Unfortunately, spam detection studies with machine learning for spam drift problem as in Twitter’s unpredictable features of newly posted tweets are mostly not satisfactory.

² <http://neuralnetworksanddeeplearning.com>

By using a deep learning technique like creating word vectors with Word2Vec tool, they have achieved higher success ratios [26].

In [27], the author introduced a deep learning technique with using SVM. The author proposes to train all layers of the deep networks by back propagating gradients through the top level of SVM, learning features of all layers. In [28], the authors demonstrate the Neural Network-based classification techniques to classify e-mails by using the initial 6,656 benign emails, 7,714 phishing emails dataset and obtain the results with 89.9% and 94.4% accuracies, and 10.1% and 5.6% inaccuracies for benign and phishing email classifications respectively. The overall accuracies and inaccuracies resulted in 92.2% and 7.8% respectively.

Recurrent Neural Networks (RNN) could be explained as deterministic transitions between hidden layers in deep learning. LSTM (Long Short-Term Memory) is considered complicated, because in addition to RNN procedure, it allows us to use memory data in timestamps. In [29], the authors put a solid RNN work with LSTM units. The authors have made Machine Translation between English and French Languages by using 160K English words and 80K French words and got higher BLEU scores. In [30], the authors have used multilayer perceptron algorithm and obtained accuracies up to 99.4%. In [31], a proposal of Deep Learning algorithm for the problem of domain adaptation of sentiment classifiers is made. SVM and Multi-Layer Perceptron (MLP) algorithms, a feed-forward type ANN, have been applied as well. In [32], authors have used Restricted Boltzmann Machine and Deep Belief Network with three different data sets, which are MNIST data set, containing 60k training and 10k testing samples of 28x28 grayscale handwritten digits [33], InfiniteMNIST, extended form of MNIST, which contains samples collected by performing random elastic forms of the real MNIST digits [34] and Shaperset, which is composed of 50000 training, 10000 validations and 10000 test images. We also used MLP and Logistic Regression in this work.

CHAPTER 3

DEEP LEARNING-BASED SPAM CLASSIFICATION

Spam email classification problem is studied from a number of different perspectives. Here, our approach is to use the latest approach in the line of developments. We use machine learning and specifically “deep learning” methods to classify emails. For this we first need to find a good email representation method that will numerically represent email instances, which is a “data representation” method. There are numerous methods in the literature. We have chosen the vector representation and tried a number of different methods to represent emails as vector values in some space. Below we first explain the data representation methods we have chosen. Then we present the deep learning methods that we decided to use to classify emails (or their vectors).

3.1 Data Representation

To be able to classify email as spam or not, we needed to first find a good data representation method that will be able to represent email as numerical values or features so that machine learning algorithms can calculate the similarities between email pairs or groups of emails. Most of the literature utilizes vector representations for emails or text as in text categorization problems. A vector represents an email as a set of numerical values, where each value is mapping to a specific feature about the email text or other features stored in e-mails (like URLs or images). In this thesis, we have studied three different vector representations for email text. We use only text

part of emails in this study. These vector representations are: (1) a weighted vector representation that we developed based on the term frequencies (TF-IDF) in email text, (2) a standard vector representation, again based on TF-IDF and used by the popular SciKit learning library for Python, (3) a recent popular approach for text vector representation approach called Word2Vec. Below we explain each vector representation.

3.1.1 Weighted TF-IDF Vectorization

Term Frequency - Inverse Document Frequency (TF-IDF) could be described as the frequency and importance of a word in a document space. This metric is frequently used in data mining and text mining tasks, or other natural language processing problems.

To calculate TF-IDF value w_d of a word w in document d , the following formula is used [35]:

$$w_d = f_{w,d} \cdot \log (|D|/f_{w,D})$$

where $f_{w,d}$ denotes the number of times word w appears in document d , that is TF, $|D|$ is the number of document in the set D , which denotes to the size of the corpus, and $f_{w,D}$ denotes the number of documents in D , in which word w appears. Logarithmic part the formula is the IDF which represents the relative frequency of a word in all documents.

We developed the following algorithm to calculate the TF-IDF based vector representation for email text in a corpus C :

- Calculate $dict_ham = \{ (w, w_d) \mid \text{TF-IDF score } w_d \text{ for each word } w \text{ in each } \mathbf{ham} \text{ document } d \text{ in the corpus } C \}$
- Calculate $dict_spam = \{ (w, w_d) \mid \text{TF-IDF score } w_d \text{ for each word } w \text{ in each } \mathbf{spam} \text{ document } d \text{ in the corpus } C \}$
- $dict_common = \{ (w, s) \mid \text{where:}$
 $s = hs - ss \text{ for each } (w, hs) \in dict_ham \text{ and } (w, ss) \in dict_spam;$
 $s = - ss \text{ for each } (w, hs) \notin dict_ham \text{ and } (w, ss) \in dict_spam;$
 $s = hs \text{ for each } (w, hs) \in dict_ham \text{ and } (w, ss) \notin dict_spam \}$

- $dict_vector \leftarrow$ sort $dict_common$, take the highest 150 and lowest 150 words from the list

$dict_vector$ now has a list of the most representative 150 words from the ham email texts and 150 words from the spam email texts. We then create a vector for each email using the words from $dict_vector$ corresponding to the frequencies of those words in each email text. We assume that if the email has more representative words from the spam words list, then the email can be considered spam.

3.1.2 TF-IDF using SciKit Learn

SciKit Learn is an open source machine learning tool, which is capable of data mining and data analysis. This tool provides supervised and unsupervised learning methods with a wide selection of parameter setting options. It is possible to represent text data as word vectors and execute classification algorithms on them [36]. SciKit learning tool requires the following [37]:

1. Python (≥ 2.7 or ≥ 3.4)
2. NumPy ($\geq 1.8.2$)
3. SciPy ($\geq 0.13.3$)

We then use Gradient Boosting (GB) to classify email vectors. GB is a machine learning technique, which is based on regression trees and found by Friedman in 1999.

3.1.3 Word2Vec

Word2Vec is a tool, which converts words from a corpus text into word vectors to represent the semantic relationships between words, such as words being used together. It is used in many recent natural language processing tasks [38]. To give an example, considering the word “Sweden” the country as an input, Word2Vec provides us a vector of words and their distances to Sweden as shown in Figure 3.

Word	Cosine distance
norway	0.760124
denmark	0.715460
finland	0.620022
switzerland	0.588132
belgium	0.585835
netherlands	0.574631
iceland	0.562368
estonia	0.547621
slovenia	0.531408

Figure 3- Vectorization of words in Word2Vec [39]

We ran Word2Vec on our email corpus sets to create word vectors for each email in the corpus.

3.2 Machine Learning Based Classification

In this thesis we used machine learning and deep learning classification algorithms for spam detection. These are the newest approaches with higher success rates. We used one of the best performing standard machine learning algorithms, namely Naive Bayes (NB) algorithm, for comparative reasons. We then used Multilayer Perceptron (MLP) and Logistic Regression in deep learning and compared the results with NB results.

Naive Bayes (NB) is a learning algorithm, which is based on the assumptions by using attributes to make an independent prediction.

$$P(c | x) = P(x | c) P(c) / P(x)$$

In the probability formula for Naive Bayes $P(c|x)$ denotes to posterior probability of class (target) given predictor (attribute), $P(c)$ is the prior probability of class, $P(x|c)$ denotes the likelihood that is the probability of predictor given class and $P(x)$ denotes to prior probability of predictor [42].

3.3 Deep Learning-Based Classification

In this section, we defined the deep learning-based classification algorithms we applied in our work.

3.3.1 Multilayer Perceptron

Multilayer Perceptron (MLP) is a neural network algorithm that contains hidden layers between input and output layers to make classification. It is a feed-forward ANN with at least 3 layers. MLP utilizes supervised learning with back propagation for training. Following figure shows the working principle of this algorithm.

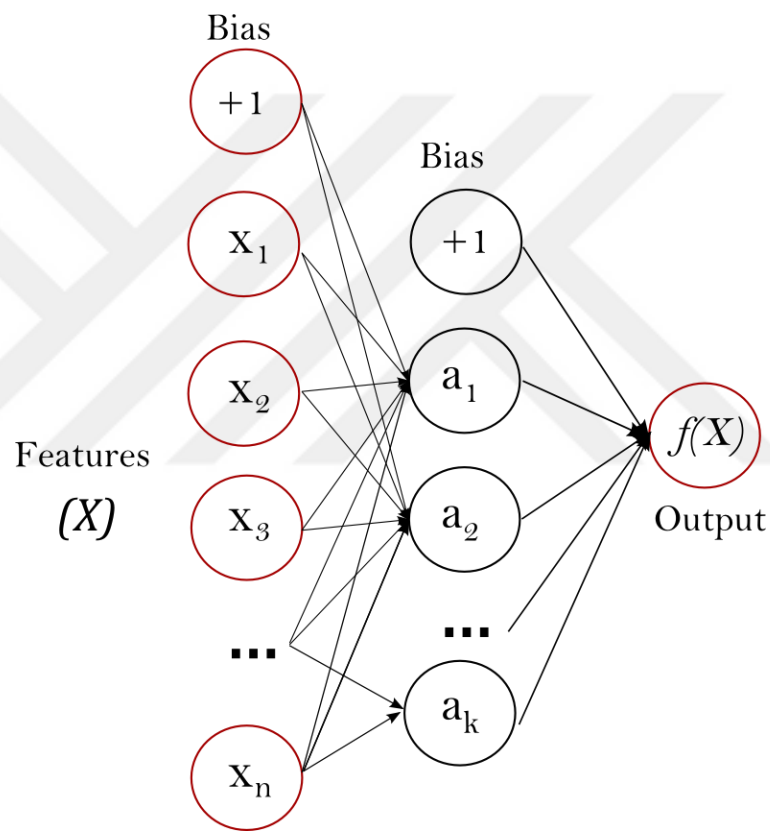


Figure 4- Multilayer Perceptron with 1-hidden layer³

In Figure 4, $[x_1, x_2, \dots, x_n]$ denotes the input features. The hidden layer neurons transform their input from the previous layer with a linear function $w_1x_1 + w_2x_2 + \dots + w_nx_n$ to the next layer [43].

³ http://scikit-learn.org/stable/modules/neural_networks_supervised.html

3.3.2 Logistic Regression

Logistic Regression is a simple classification algorithm, especially to make binary classifications. Our spam detection problem is therefore a perfect candidate for this algorithm⁴ [44]. The binary logistic model is used for the probability estimation of a two-class response (like ham/spam) based on the input feature values [45].

3.3.3 Keras on TensorFlow

We used TensorFlow⁵ to apply deep learning methods on the vector representations. Tensorflow, which is an open source software library, provides us numerical computation by using data flow graphs, implementing deep learning architectures such as Recurrent Neural Networks (RNN) and Deep Neural Networks (DNN). In this graph data structure, nodes are equal to mathematical operations and graph edges are equal to multidimensional data arrays (tensors) and they communicate among themselves [40].

Besides TensorFlow, Keras is also another library and a neural network API, which has an ability to work on TensorFlow, CNTK, and Theano as well. Keras is suitable to work with Python 2.7 and upper versions [41].

⁴ <http://ufldl.stanford.edu/tutorial/supervised/LogisticRegression>

⁵ <http://www.tensorflow.org>

CHAPTER 4

EVALUATION

In this chapter we present the evaluation results. We first explain the datasets we used in the evaluation. Then the results are presented and analyzed.

4.1 Datasets

For our early tests, we used Comodemia's Phishing E-mail Set⁶, which is an open dataset from Comodo Academy [46]. But it is limited in size and context, including only 100 phishing emails and no ham emails. Therefore, later we started to use another open dataset from the community, Enron dataset. Enron E-mail Dataset⁷ is relatively larger in size and context, including 10K ham and spam e-mails. Enron Corporation is a bankrupted American company, which was a major company in the areas of energy, commodities, and services in the United States [47]. After bankruptcy, organization's all secret and commercial company e-mails spread around the Internet. We used mainly this dataset for testing out methods.

As we pointed out above, we used only the textual data in emails and specifically email subject and email body text. Other email parameters are not considered. Enron emails includes only the textual data; therefore, parsing was easier.

⁶ Comodemia, Comodo Academy,
https://comodemia.comodo.com/here_is_what_you_get.php

⁷ Enron Dataset of Carnegie Mellon University, School of Computer Science
<https://www.cs.cmu.edu/~enron/>

We sampled the Enron dataset in increasing sizes and tested these subsamples to see the effect of corpus size in the experiments. We also sampled ham and spam emails in equal sizes to generate the datasets.

We created word vectors using top 300 most frequently used words (attribute size = 300). Table 2 shows the word vector files generated from ham and spam e-mails and their sizes regarding the vector generation methods Word2Vec and Weighted TF-IDF. Word2Vec vectors are larger as expected compared to the TF-IDF vectors.

Table 2- Vector names, generation methods and their sizes

Dataset	Vector Method	Size (MB)
300 ham + 300 spam	Word2Vec	1.3
500 ham + 500 spam	Word2Vec	2.2
1k ham + 1k spam	Word2Vec	4.4
2k ham + 2k spam	Word2Vec	8.9
5k ham + 5k spam	Word2Vec	22.2
10k ham + 10k spam	Word2Vec	44.4
300 ham + 300 spam	Weighted TF-IDF	0.9
500 ham + 500 spam	Weighted TF-IDF	1.5
1k ham + 1k spam	Weighted TF-IDF	3
2k ham + 2k spam	Weighted TF-IDF	6
5k ham + 5k spam	Weighted TF-IDF	15
10k ham + 10k spam	Weighted TF-IDF	30

4.2 Tools and Libraries

We used following tools, libraries, programming languages and applications to evaluate the dataset we used and analyze the results.

For development, we used PyCharm 2017.3.3 and Visual Studio Code 1.20.1 as IDE / Text Editor. As programming languages, we used only Python and its version 3.5m and 2.7.

As deep learning tools, we used for using vectors as input and algorithms as output are as follow: TensorFlow⁸ 1.0, Keras 2.1.5⁹, SciKit Learn¹⁰ 0.19.1

We also used numerous libraries such as NLTK 3.2.5, Gensim 3.4.0, PIP 9.0.1 and NumPy 1.14.0 during development phase before vector generation. We used Gensim to apply Word2Vec, NumPy to make scientific calculations in Python, PIP to manage installed packages and install package in Python and NLTK to parse, and semantic reason, wrap for texts in Python.

To execute the ML and DL Algorithms as Machine Learning Application with GUI, we also used WEKA¹¹ tool with the following components:

- WEKA 3.6.3, GUI Application
- WEKA 3.9.2 Developer Edition

4.3 Evaluation Metrics

F-measure is used for the evaluation of tests. For F-measure scores, the following scores are calculated: True Positive, True Negative, False Positive and False Negative.

- True Positive (TP): Refers to the test result that defines a given condition exists, when it does.
- True Negative (TN): Refers to test result that defines that a condition does not take place, although in fact it does not.
- False Positive (FP): Refers to test result that defines a given condition exists, although it does not.

⁸ Tensorflow, <https://www.tensorflow.org/>

⁹ Keras, <https://keras.io/>

¹⁰ SciKit Learn, <http://scikit-learn.org/>

¹¹ WEKA, <https://www.cs.waikato.ac.nz/ml/weka/>

- False Negative (FN): Refers to test result that defines that a condition does not take place, although in fact it does.

The following table summarized the relationships between these scores on the observations of the results.

Table 3- Relationship between scores on results

Actual Class	Predicted class	
	Class = Yes	Class = No
Class = Yes	True Positive	False Negative
Class = No	False Positive	True Negative

Accuracy, precision and F-measures are then calculated as follows:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN}$$

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$F1\ Score = 2 * (\frac{Recall * Precision}{Recall + Precision})$$

Precision measures the ratio of correctly predicted positive results to the total positive predictions. Recall is the ratio of correctly predicted results to all observations in a specific class. F-measure is then a weighted average of Precision and Recall.

4.3.1 Test Plans

During learning/testing process, we tested the following test options as provided in WEKA tool:

- **66 Percentage:** In this test option, 66% of data (emails) are used for training the model and 34% is used for testing (validation).
- **80 Percentage:** In this test option, 80% of data is used training and 20% is used for testing.

- **K-Folds:** In this test option, data is divided into k equal parts. For each part, the other parts ($k - 1$ parts) are used for training and the selected part is used for testing. The average of all k test results is calculated as the result of testing. We set k to 10 and hence 10% of dataset is used for testing in every fold.

4.3.2 Results

We tested with several different train/test ratios on the datasets. These are explained below.

4.3.2.1 Results with Weighted TF-IDF Vector Representation

4.3.2.1.1 WEKA Results

Accuracy results for all datasets and test methods using WEKA tool and 3 different ML algorithms (NB, MLP, LR) are shown in Table 4.

Table 4- Results for Weighted TF-IDF method on WEKA

Data Sets (Ham + Spam)	Process	Naive Bayes (%)	Multilayer Perceptron (%)	Logistic Regression (%)
300+300	10 Folds	62	54	61.83
	%66 Percentage	61.76	63.23	68.62
	%80 Percentage	65	45.83	63.33
500+500	10 Folds	59.9	50.8	59.2
	%66 Percentage	61.47	54.41	55.58
	%80 Percentage	63.5	44.5	50.5
1000+1000	10 Folds	53.1	50.1	57.1
	%66 Percentage	54.55	52.2	58.23
	%80 Percentage	52.5	50.25	56.25
2000+2000	10 Folds	53.45	50.12	54.75
	%66 Percentage	54.92	50.8	55.36
	%80 Percentage	52.25	48.37	52.5
5000+5000	10 Folds	53.12	50.01	53.29
	%66 Percentage	53.17	50.17	54.08
	%80 Percentage	53.45	49.9	53.8
10000+10000	10 Folds	50.38	50	52.98
	%66 Percentage	50.75	50.08	52.83
	%80 Percentage	50.05	50.47	53.52

The accuracy results in Table 3 are shown as bar graphs for comparison in the following figures (Figure 5-10 for datasets 300+300, 500+500, 1k+1k, 2k+2k, 5k+5k, 10k+10k).

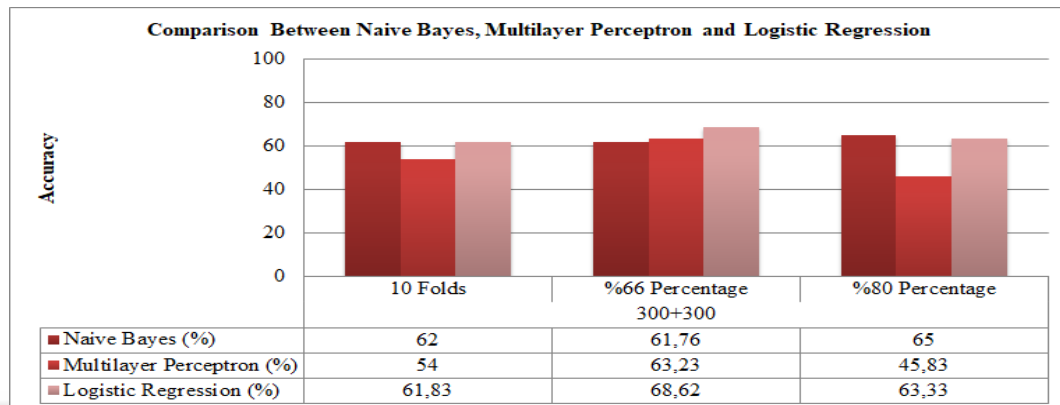


Figure 5- Accuracy Comparison of Weighted TF-IDF method algorithms for 300 ham + 300 spam datasets.

As seen on Figure 5, for 10 Folds method, best result is obtained by using Naive Bayes algorithm. For 66% method, best score is obtained by Logistic Regression and for 80% method, best score is obtained using Naive Bayes with 300 ham + 300 spam data.

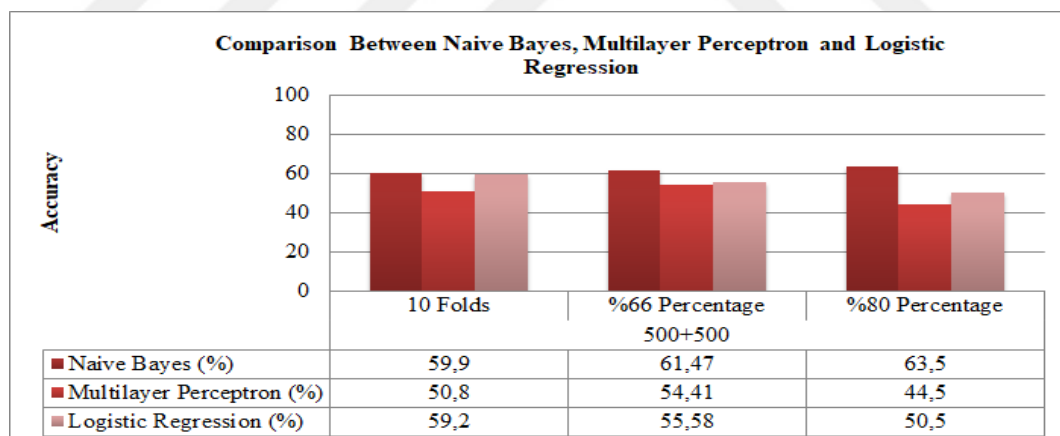


Figure 6- Accuracy Comparison of Weighted TF-IDF method algorithms for 500 ham + 500 spam datasets.

As seen on Figure 6, for 10 Folds method, best result is obtained by using Naive Bayes algorithm. For 66% method, best score is obtained by Naive Bayes and for 80% method, best score is obtained by Naive Bayes with 500 ham + 500 spam data. Naive Bayes performs the best in all three test cases.

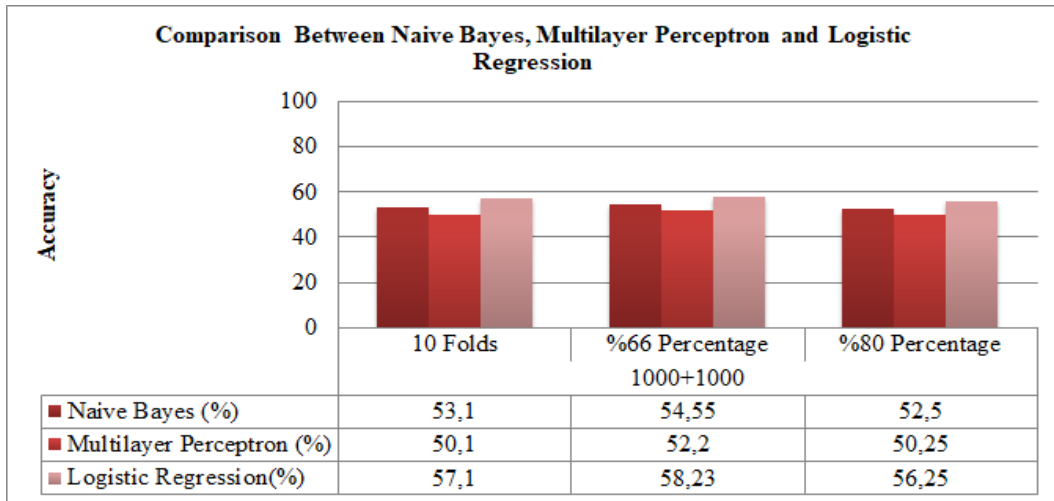


Figure 7- Accuracy Comparison of Weighted TF-IDF method algorithms for 1000 ham + 1000 spam dataset.

As seen on Figure 7, for 10 Folds method on 1000+1000 dataset, the best result is obtained by using Logistic Regression algorithm. For 66% method, best score is obtained by Logistic Regression and for 80% method, best score is obtained by Logistic Regression with 1000 ham + 1000 spam data. In all test cases Logistic Regression performs the best.

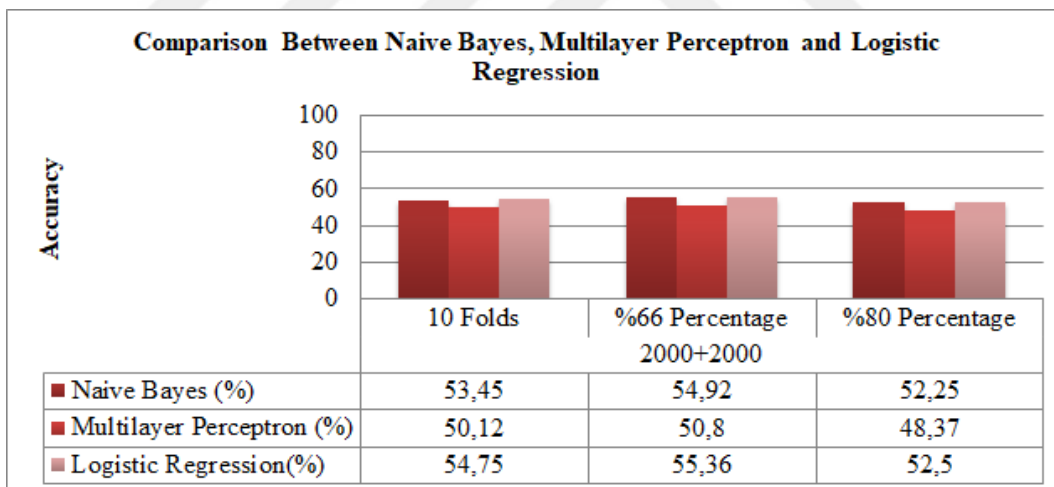


Figure 8- Accuracy Comparison of Weighted TF-IDF method algorithms for 2000 ham + 2000 spam dataset.

Figure 8 shows the results for 2000+2000 dataset, Figure 9 shows the results for 5k+5k dataset and Figure 10 shows the results for 10k+10k dataset. In all cases and in all test cases Logistic Regression performs the best against Naïve Bayes and MLP. We expected LR to perform the best in binary classification with limited datasets.

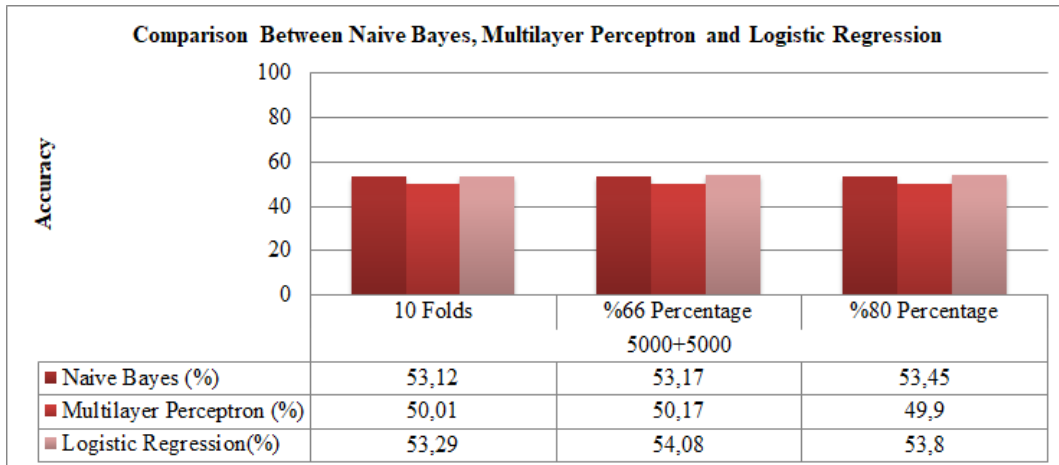


Figure 9- Accuracy Comparison of Weighted TF-IDF method algorithms for 5000 ham + 5000 spam datasets.

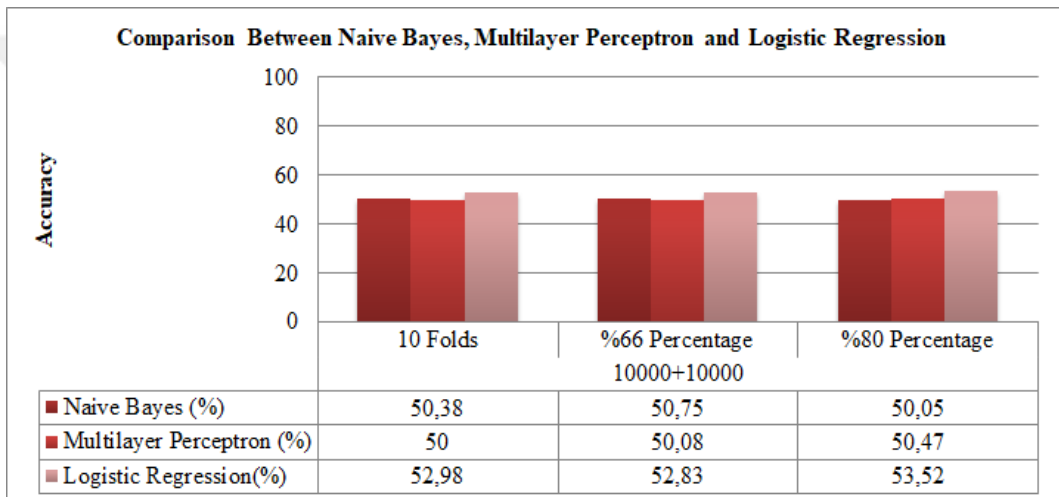


Figure 10- Accuracy Comparison of Weighted TF-IDF method algorithms for 10000 ham + 10000 spam datasets.

Table 5 lists the F-measure results for datasets 300+300 and 500+500.

Table 5 – F Measure Results for 300 + 300 and 500 + 500 ham – spam datasets

Data Sets	Test Type	Naive Bayes				Logistic Regression				Multilayer Perceptron			
300+300	Cross Validation (10 Folds)	SPAM	HAM		0.620	SPAM	HAM		0.618	SPAM	HAM		0.540
		240	60	SPAM		145	155	SPAM		132	168	SPAM	
		168	132	HAM		74	226	HAM		108	192	HAM	
	Percentage (%66)	SPAM	HAM		0.618	SPAM	HAM		0.686	SPAM	HAM		0.632
		82	18	SPAM		55	45	SPAM		31	69	SPAM	
		60	44	HAM		19	85	HAM		6	98	HAM	
	Percentage (%80)	SPAM	HAM		0.650	SPAM	HAM		0.625	SPAM	HAM		0.458
		46	10	SPAM		23	33	SPAM		54	2	SPAM	
		32	32	HAM		11	53	HAM		63	1	HAM	
500+500	Cross Validation (10 Folds)	SPAM	HAM		0.599	SPAM	HAM		0.592	SPAM	HAM		0.508
		452	48	SPAM		187	313	SPAM		247	253	SPAM	
		353	147	HAM		95	405	HAM		239	261	HAM	
	Percentage (%66)	SPAM	HAM		0.615	SPAM	HAM		0.556	SPAM	HAM		0.544
		171	12	SPAM		67	116	SPAM		182	1	SPAM	
		119	38	HAM		35	122	HAM		154	3	HAM	
	Percentage (%80)	SPAM	HAM		0.635	SPAM	HAM		0.505	SPAM	HAM		0.445
		103	8	SPAM		38	73	SPAM		0	111	SPAM	
		65	24	HAM		26	63	HAM		0	89	HAM	

Table 6 lists the F-measure results for datasets 1k+1k and 2k+2k.

Table 6– F Measure Results for 1000 + 1000 and 2000 + 2000 ham – spam datasets

Data Sets	Test Type	Naive Bayes				Logistic Regression				Multilayer Perceptron			
1000+1000	Cross Validation (10 Folds)	SPAM	HAM		0.531	SPAM	HAM		0.571	SPAM	HAM		0.501
		79	921	SPAM		267	733	SPAM		398	602	SPAM	
		17	983	HAM		125	875	HAM		396	604	HAM	
	Percentage (%66)	SPAM	HAM		0.546	SPAM	HAM		0.582	SPAM	HAM		0.522
		40	301	SPAM		95	246	SPAM		316	25	SPAM	
		8	331	HAM		38	301	HAM		300	39	HAM	
	Percentage (%80)	SPAM	HAM		0.525	SPAM	HAM		0.563	SPAM	HAM		0.503
		15	185	SPAM		53	147	SPAM		200	0	SPAM	
		5	195	HAM		28	172	HAM		199	1	HAM	
2000+2000	Cross Validation (10 Folds)	SPAM	HAM		0.535	SPAM	HAM		0.548	SPAM	HAM		0.501
		998	1002	SPAM		340	1660	SPAM		1199	801	SPAM	
		860	1140	HAM		150	1850	HAM		1194	806	HAM	
	Percentage (%66)	SPAM	HAM		0.549	SPAM	HAM		0.554	SPAM	HAM		0.508
		655	28	SPAM		118	565	SPAM		660	23	SPAM	
		585	92	HAM		42	635	HAM		646	31	HAM	
	Percentage (%80)	SPAM	HAM		0.523	SPAM	HAM		0.525	SPAM	HAM		0.484
		346	41	SPAM		345	42	SPAM		387	0	SPAM	
		341	72	HAM		338	75	HAM		413	0	HAM	

Table 7 lists the F-measure results for datasets 5k+5k and 10k+10k.

Table 7– F Measure Results for 5000 + 5000 and 10000 + 10000 ham – spam datasets

#Data Sets	Test Type	Naive Bayes				Logistic Regression				Multilayer Perceptron			
5000+5000	Cross Validation (10 Folds)	SPAM	HAM		0.531	SPAM	HAM		0.533	SPAM	HAM		0.500
		4968	32	SPAM		4787	213	SPAM		1000	4000	SPAM	
		4656	344	HAM		4458	542	HAM		999	4001	HAM	
	Percentage (%66)	SPAM	HAM		0.318	SPAM	HAM		0.541	SPAM	HAM		0.502
		1679	15	SPAM		1645	49	SPAM		0	1674	SPAM	
		1577	129	HAM		1512	194	HAM		0	1706	HAM	
	Percentage (%80)	SPAM	HAM		0.535	SPAM	HAM		0.538	SPAM	HAM		0.499
		990	12	SPAM		961	41	SPAM		0	1002	SPAM	
		919	79	HAM		883	115	HAM		0	998	HAM	
10000+10000	Cross Validation (10 Folds)	SPAM	HAM		0.504	SPAM	HAM		0.530	SPAM	HAM		0.500
		121	9879	SPAM		9754	246	SPAM		5000	5000	SPAM	
		44	9956	HAM		9157	843	HAM		5000	5000	HAM	
	Percentage (%66)	SPAM	HAM		0.508	SPAM	HAM		0.528	SPAM	HAM		0.501
		63	3331	SPAM		3316	78	SPAM		0	3394	SPAM	
		18	3388	HAM		3129	277	HAM		0	3406	HAM	
	Percentage (%80)	SPAM	HAM		0.501	SPAM	HAM		0.535	SPAM	HAM		0.505
		29	1990	SPAM		1972	47	SPAM		2018	1	SPAM	
		8	1973	HAM		1812	169	HAM		1980	1	HAM	

F-Measure comparisons for Weighted TF-IDF data representation and WEKA tests for all algorithms are shown in Figure 11-16. There are varying results here. For example, for 300+300 dataset (Figure 11), NB performs better in 10-fold test, LR performs better in 66% test, and NB performs again better in 80% test. But as the dataset gets larger (to 10k+10k) it is clear LR performs better than NB and MLP (Figure 12-16).

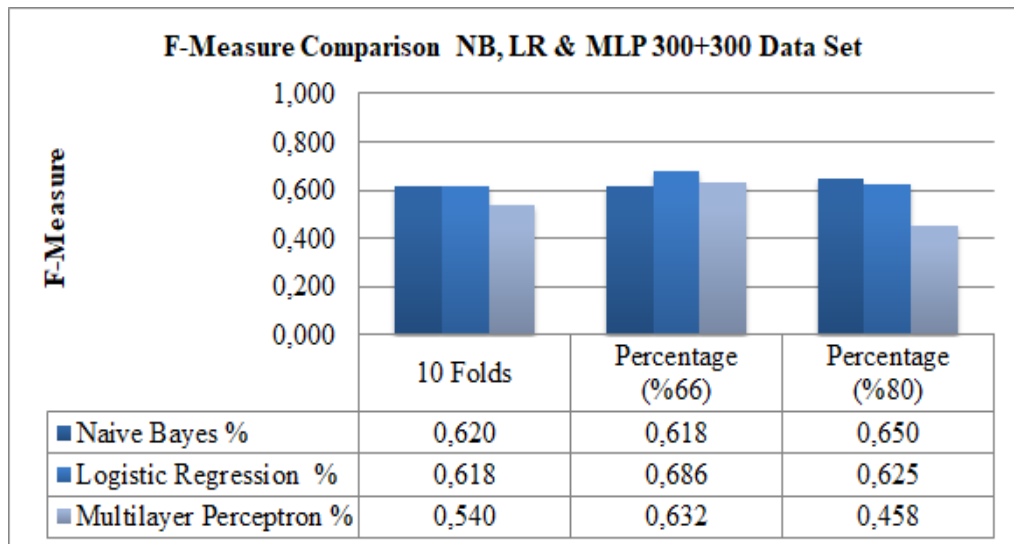


Figure 11- Algorithm Comparison for 300 + 300 Data Set

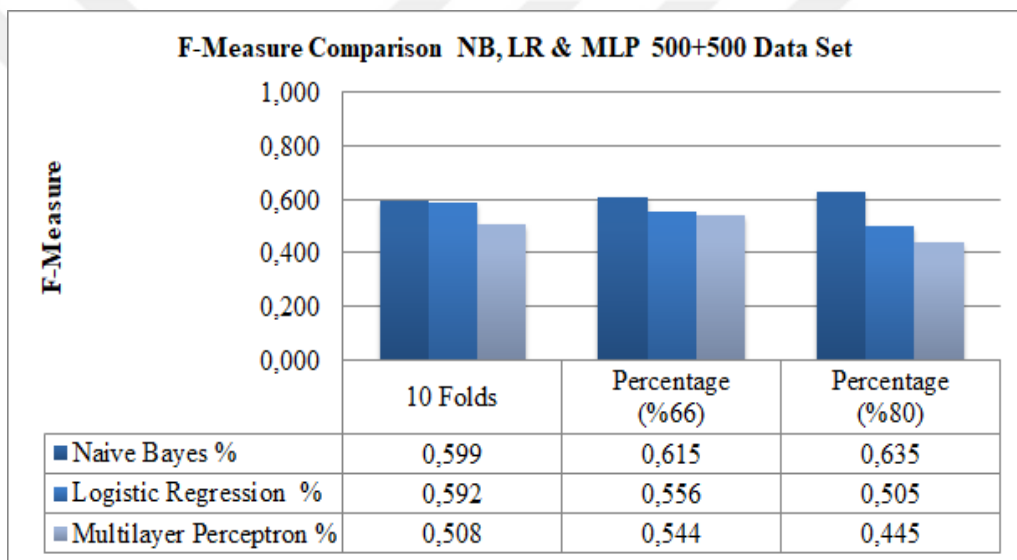


Figure 12- Algorithm Comparison for 300 + 300 Data Set

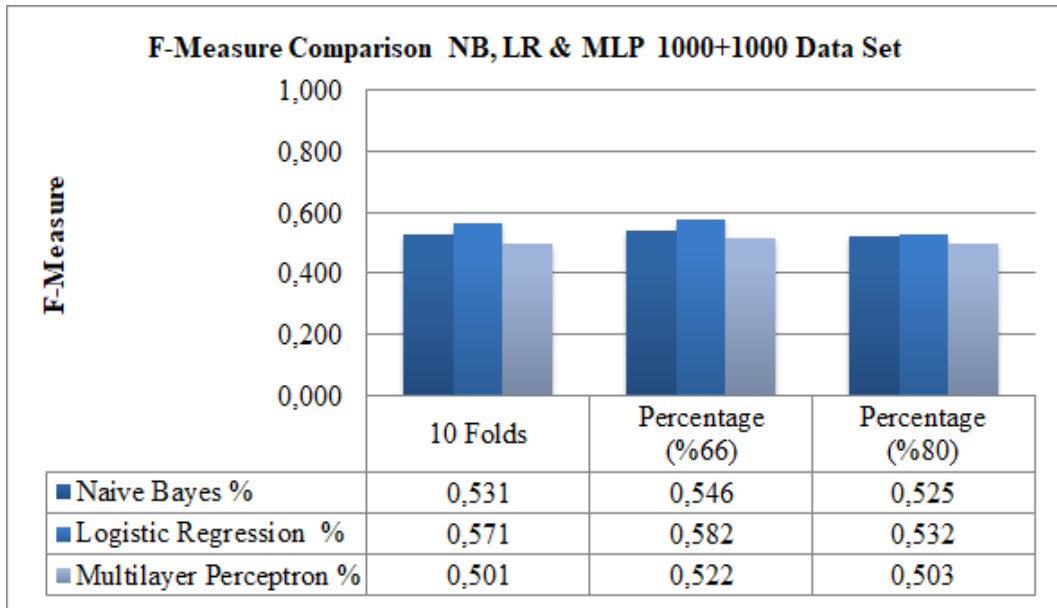


Figure 13 - Algorithm Comparison for 1000 + 1000 Data Set

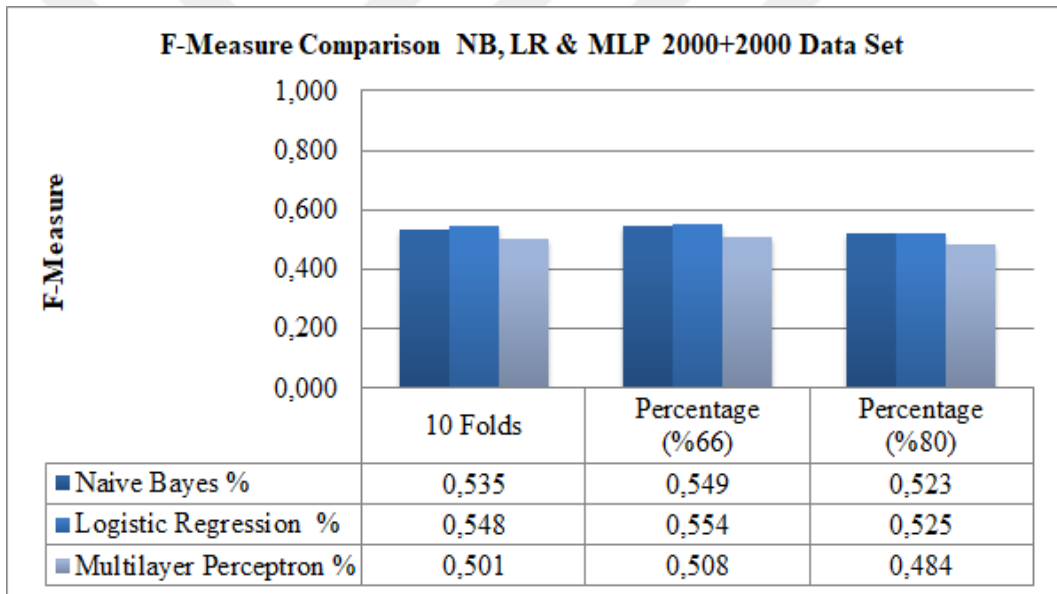


Figure 14- Algorithm Comparison for 2000 + 2000 Data Set

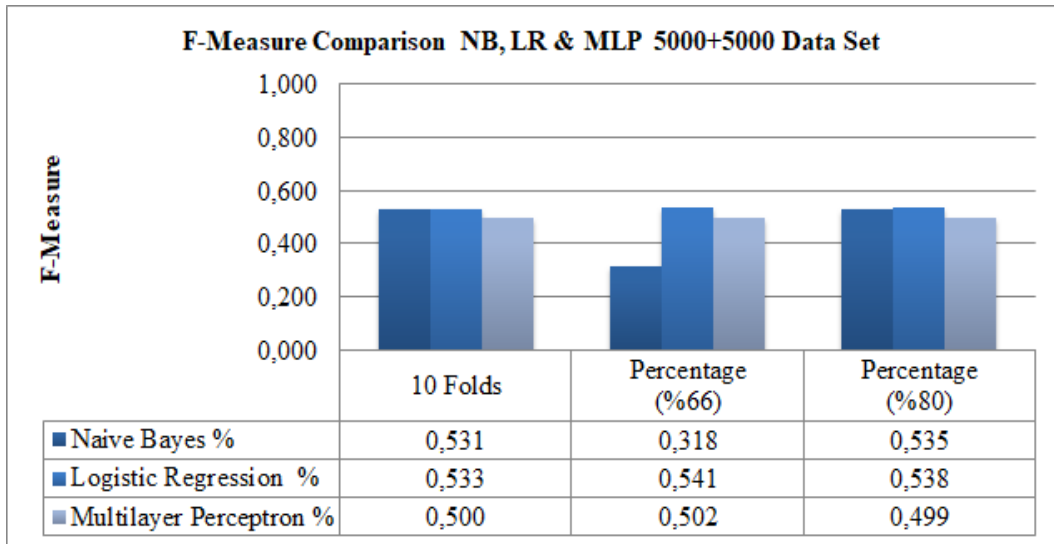


Figure 15- Algorithm Comparison for 5000 + 5000 Data Set

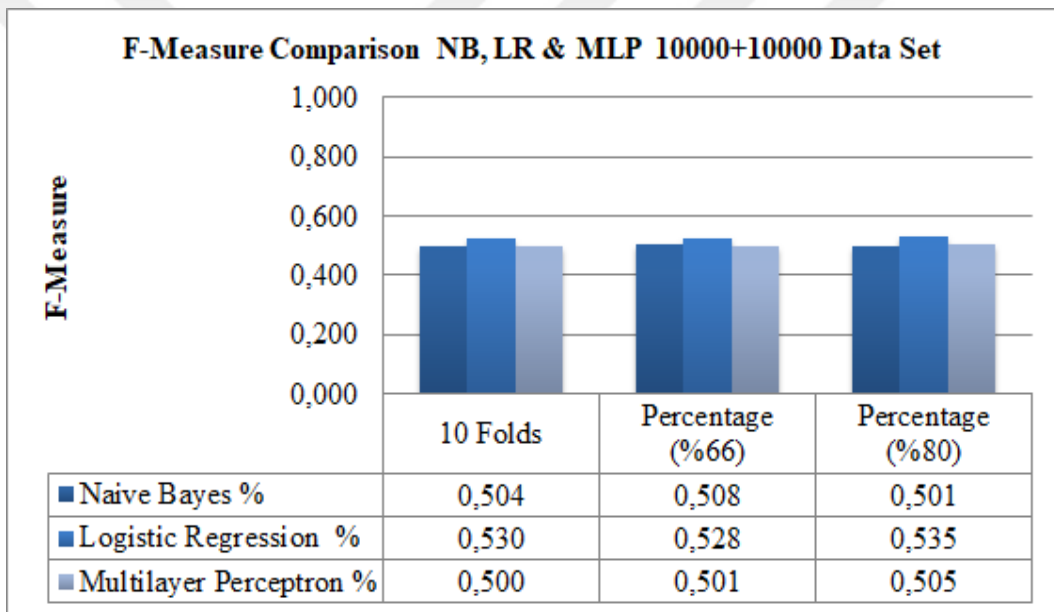


Figure 16- Algorithm Comparison for 10000 + 10000 Data Set

4.3.2.1.2 TensorFlow

We have also test our datasets with TensorFlow. The neural network's hidden layer is set to 10 neurons with input dimension 300 using Adam gradient optimizer (Keras interface). The results show that as the dataset gets larger Tensorflow gives worse results from 63% to 53% accuracy levels (Figure 17 and Table 7).

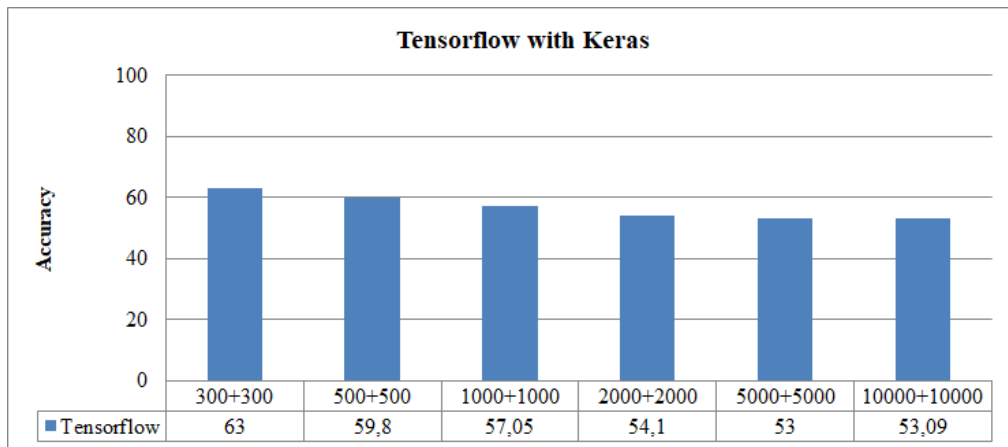


Figure 17- TensorFlow with Keras for all data sets

10-Fold Cross Validation Results for Weighted TF-IDF is as follows:

Table 8– 10-Fold Cross Validation Results in TensorFlow + Keras

Data Sets (Ham + Spam)	Cross Validation 10-Fold
300+300	63
500+500	59.8
1000+1000	57.05
2000+2000	54.1
5000+5000	53
10000+10000	53.09

4.3.2.3 Results with SciKit Learn TF-IDF Vector Representation

By using SciKit Learn tool, we obtained the following results with SVM-based classification.

Table 9- Test Results for SciKit Learn

Data Set (Ham + Spam)	SciKit Learn
300	92
500	93
1000	93.25
2000	95.33
5000	96.31
10000	96.845

Accuracy result graphics for SciKit Learn are shown in Figure 18 for all data sets.

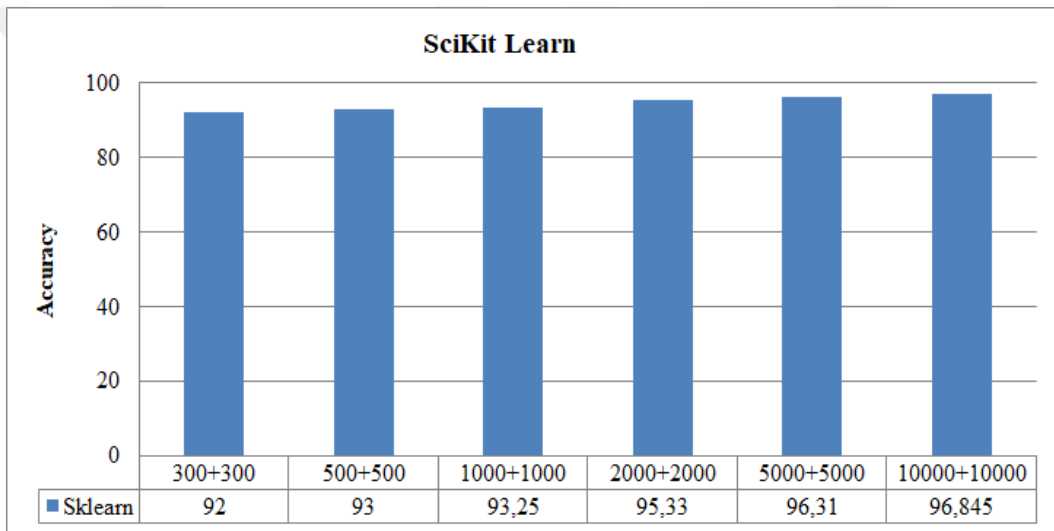


Figure 18- Accuracy Results for SciKit Learn

4.3.2.4 Results with Word2Vec Vector Representation

Here we present the ML results for Word2Vec data representation, first with WEKA tool and then with Keras on Tensorflow.

4.3.2.4.1 WEKA Results

Following Tables demonstrates the success ratios of algorithms, Accuracy Results regarding to Word2Vec implementation. For all algorithms, best score is highlighted with bold text for each data set.

Table 10– Weka Results for Word2Vec implementation

Datasets (Ham + Spam)	Process	Naive Bayes (%)	Multilayer Perceptron (%)	Logistic Regression (%)
300+300	10 Folds	75.16	93.66	91.33
	%66 Percentage	78.92	95.58	85.78
	%80 Percentage	71.66	97.5	91.66
500+500	10 Folds	69.4	94.2	84.2
	%66 Percentage	68.52	89.7	88.82
	%80 Percentage	69	94	89.5
1000+1000	10 Folds	72.5	95.1	89.35
	%66 Percentage	70.14	96.17	85.29
	%80 Percentage	69.75	96.5	90.75
2000+2000	10 Folds	75.2	94.72	92.95
	%66 Percentage	77.5	95.66	91.61
	%80 Percentage	76.75	94.87	93.37
5000+5000	10 Folds	74.5	95.64	95.81
	%66 Percentage	74.82	96.02	95.5
	%80 Percentage	73.75	95.6	95.85
10000+10000	10 Folds	75.77	96.83	96.18
	%66 Percentage	75.63	96.39	95.88
	%80 Percentage	76.37	96.85	96.15

For all datasets MLP performs better in all 3 test plans (Figure 19-23). This was expected since Word2Vec is close to deep learning solutions as semantic relations in documents are captured in vector representation.

We could not test WEKA with 10k+10k dataset since the test taking too long and the tests crashed on the machine we were testing.

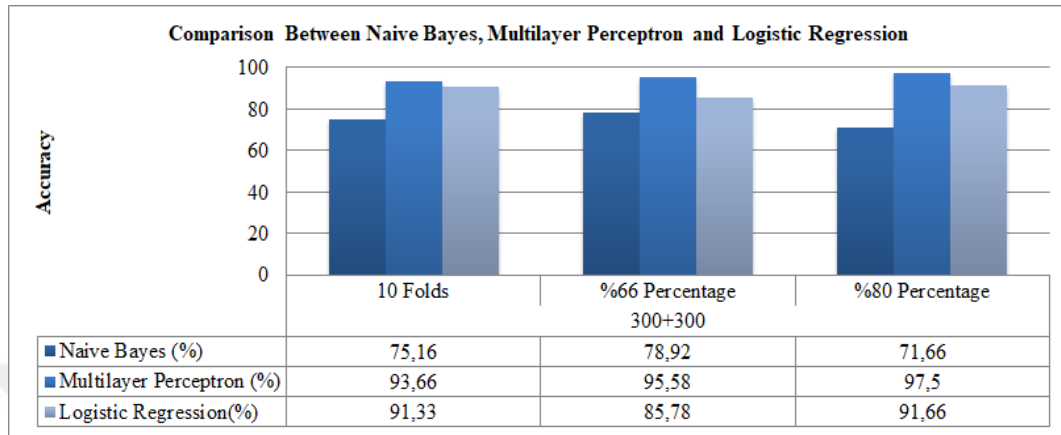


Figure 19 - Success Ratios for Word2Vec 300 + 300 data sets

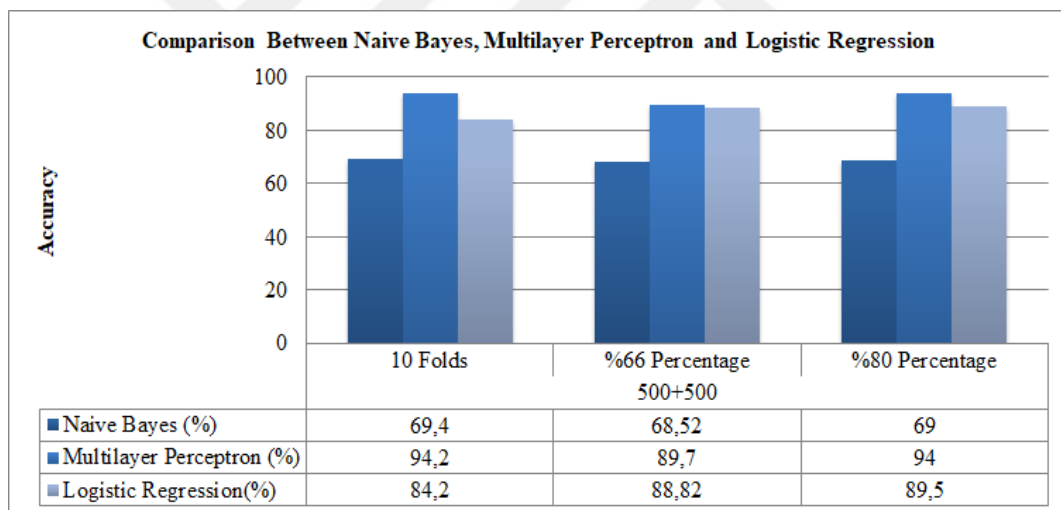


Figure 20- Success Ratios for Word2Vec 500 + 500 data sets

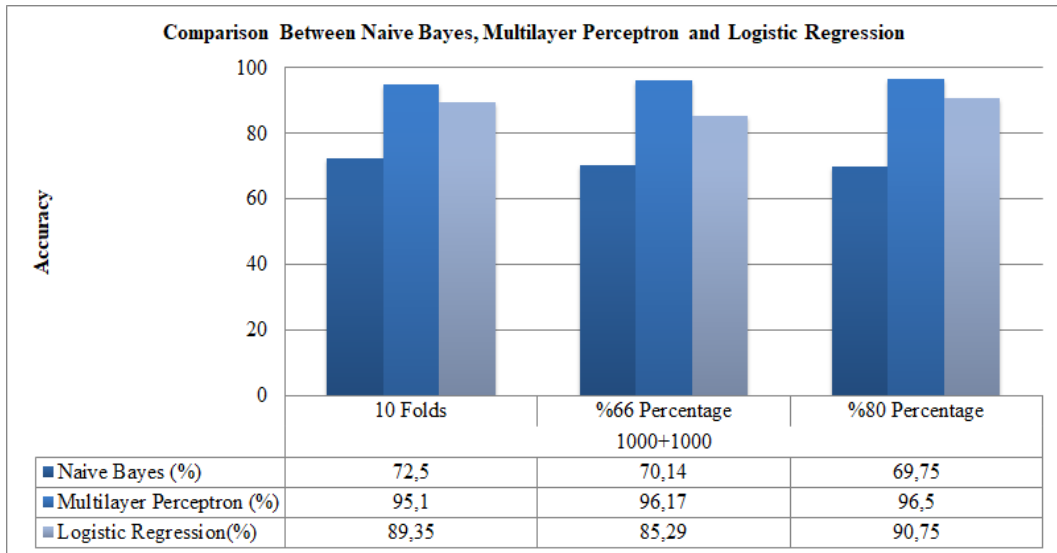


Figure 21- Success Ratios for Word2Vec 1000 + 1000 data sets

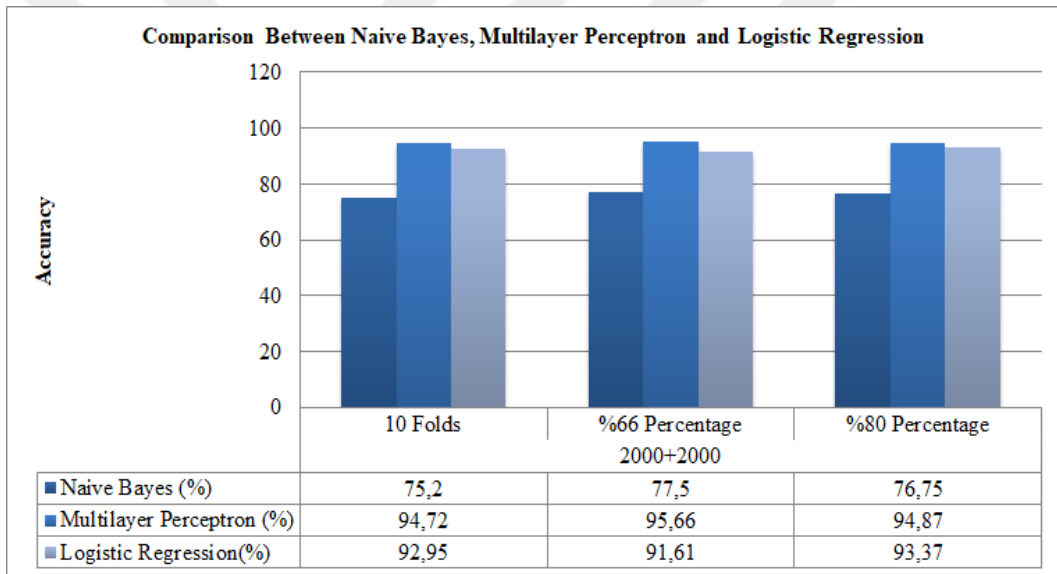


Figure 22- Success Ratios for Word2Vec 2000 + 2000 data sets

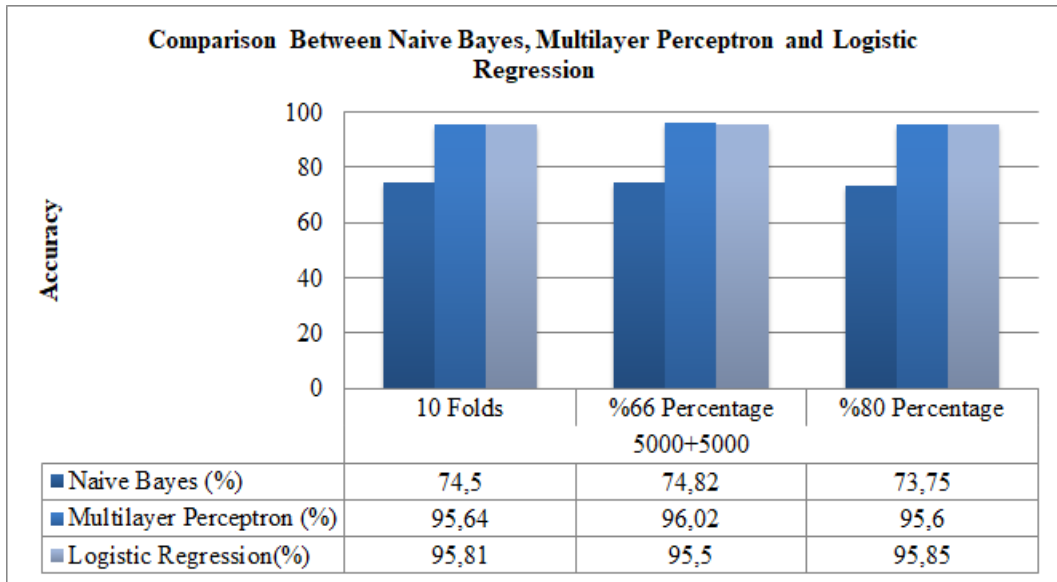


Figure 23- Success Ratios for Word2Vec 5000 + 5000 data sets

F Measure Results for Word2Vec implementation are as follow. For all algorithms, best score is highlighted with bold text for each data set:

Table 11- F Measure Results for Word2Vec 300 + 300 and 500 + 500 implementation

Data Sets	Test Type	Naive Bayes			Logistic Regression				Multilayer Perceptron				
300+300	Cross Validation (10 Folds)	SPAM	HAM		0.752	SPAM	HAM		0.913	SPAM	HAM		0.937
		187	113	SPAM		274	26	SPAM		284	16	SPAM	
		36	264	HAM		26	274	HAM		22	278	HAM	
	Percentage (%66)	SPAM	HAM		0.789	SPAM	HAM		0.858	SPAM	HAM		0.956
		65	35	SPAM		85	15	SPAM		97	3	SPAM	
		8	96	HAM		14	90	HAM		6	98	HAM	
	Percentage (%80)	SPAM	HAM		0.717	SPAM	HAM		0.917	SPAM	HAM		0.975
		31	25	SPAM		51	5	SPAM		55	1	SPAM	
		9	55	HAM		5	59	HAM		2	62	HAM	
500+500	Cross Validation (10 Folds)	SPAM	HAM		0.694	SPAM	HAM		0.842	SPAM	HAM		0.942
		276	224	SPAM		417	83	SPAM		477	23	SPAM	
		82	418	HAM		75	425	HAM		35	465	HAM	
	Percentage (%66)	SPAM	HAM		0.682	SPAM	HAM		0.888	SPAM	HAM		0.897
		95	88	SPAM		161	22	SPAM		165	18	SPAM	
		19	138	HAM		16	141	HAM		17	140	HAM	
	Percentage (%80)	SPAM	HAM		0.690	SPAM	HAM		0.895	SPAM	HAM		0.940
		58	53	SPAM		99	12	SPAM		106	5	SPAM	
		9	80	HAM		9	80	HAM		7	82	HAM	

Table 12- F Measure Results for Word2Vec 1000 + 1000 and 2000 + 2000 implementation

Data Sets	Test Type	Naive Bayes				Logistic Regression				Multilayer Perceptron			
1000+1000	Cross Validation (10 Folds)	SPAM	HAM		0.725	SPAM	HAM		0.894	SPAM	HAM		0.951
		593	407	SPAM		901	99	SPAM		955	45	SPAM	
		143	857	HAM		114	886	HAM		53	947	HAM	
	Percentage (%66)	SPAM	HAM		0.701	SPAM	HAM		0.853	SPAM	HAM		0.962
		204	137	SPAM		284	57	SPAM		327	14	SPAM	
		66	273	HAM		43	296	HAM		12	327	HAM	
	Percentage (%80)	SPAM	HAM		0.698	SPAM	HAM		0.908	SPAM	HAM		0.965
		115	85	SPAM		182	18	SPAM		192	8	SPAM	
		36	164	HAM		19	181	HAM		6	194	HAM	
2000+2000	Cross Validation (10 Folds)	SPAM	HAM		0.752	SPAM	HAM		0.930	SPAM	HAM		0.947
		1268	732	SPAM		1866	134	SPAM		1890	110	SPAM	
		260	1760	HAM		148	1852	HAM		101	1899	HAM	
	Percentage (%66)	SPAM	HAM		0.775	SPAM	HAM		0.916	SPAM	HAM		0.957
		435	248	SPAM		619	64	SPAM		659	24	SPAM	
		58	619	HAM		50	657	HAM		35	642	HAM	
	Percentage (%80)	SPAM	HAM		0.768	SPAM	HAM		0.934	SPAM	HAM		0.949
		241	146	SPAM		357	30	SPAM		376	11	SPAM	
		40	373	HAM		23	390	HAM		30	383	HAM	

Table 13- F Measure Results for Word2Vec 5000 + 5000 and 10000 + 10000 implementation

Data Sets	Test Type	Naive Bayes			Logistic Regression			Multilayer Perceptron					
5000+5000	Cross Validation (10 Folds)	SPAM	HAM			SPAM	HAM			SPAM	HAM		
		3099	1901	SPAM	0.745	4804	196	SPAM	0.958	4786	214	SPAM	0.956
		649	4351	HAM		223	4777	HAM		222	4778	HAM	
	Percentage (%66)	SPAM	HAM			SPAM	HAM			SPAM	HAM		
		1059	635	SPAM	0.748	1623	71	SPAM	0.955	1627	67	SPAM	0.960
		221	1485	HAM		82	1624	HAM		68	1638	HAM	
	Percentage (%80)	SPAM	HAM			SPAM	HAM			SPAM	HAM		
		619	383	SPAM	0.738	965	37	SPAM	0.959	970	32	SPAM	0.956
		142	856	HAM		46	952	HAM		56	942	HAM	
10000+10000	Cross Validation (10 Folds)	SPAM	HAM			SPAM	HAM			SPAM	HAM		
		6350	3650	SPAM	0.758	9670	330	SPAM	0.962	9702	298	SPAM	0.968
		1196	8804	HAM		433	9567	HAM		335	9665	HAM	
	Percentage (%66)	SPAM	HAM			SPAM	HAM			SPAM	HAM		
		5143	2136	SPAM	0.756	3277	117	SPAM	0.959	3257	137	SPAM	0.964
		6800	399	HAM		163	3243	HAM		108	3298	HAM	
	Percentage (%80)	SPAM	HAM			SPAM	HAM			SPAM	HAM		
		1304	715	SPAM	0.764	1956	63	SPAM	0.962	1970	49	SPAM	0.969
		230	1751	HAM		91	1860	HAM		77	1904	HAM	

F-measure results are also shown comparatively in Figures 24-29 for all datasets with Word2Vec representations. Clearly MLP performs better in all cases, sometimes with up to 20% better F-measure values in comparison to NB.

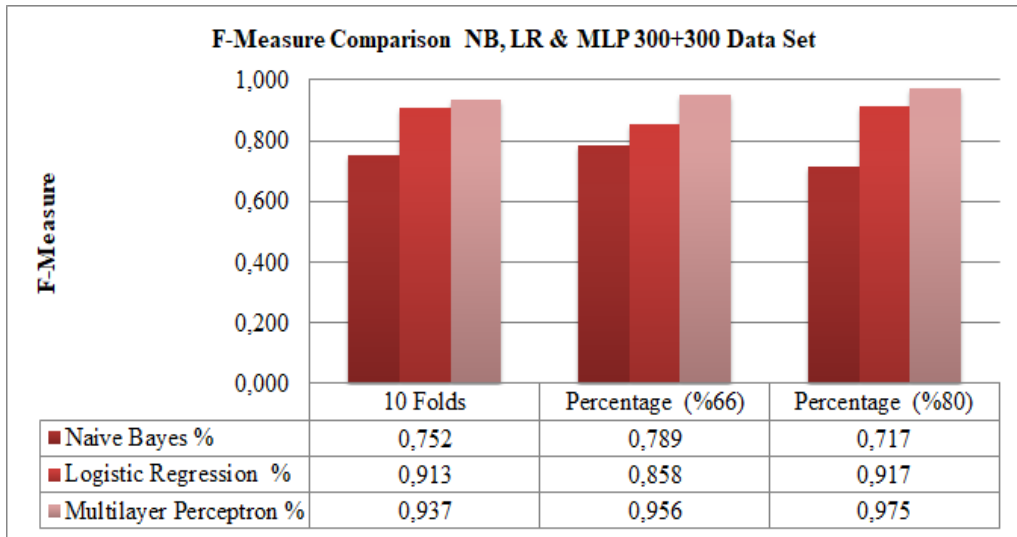


Figure 24- F Measure Graphics for Word2Vec 300 + 300 data sets

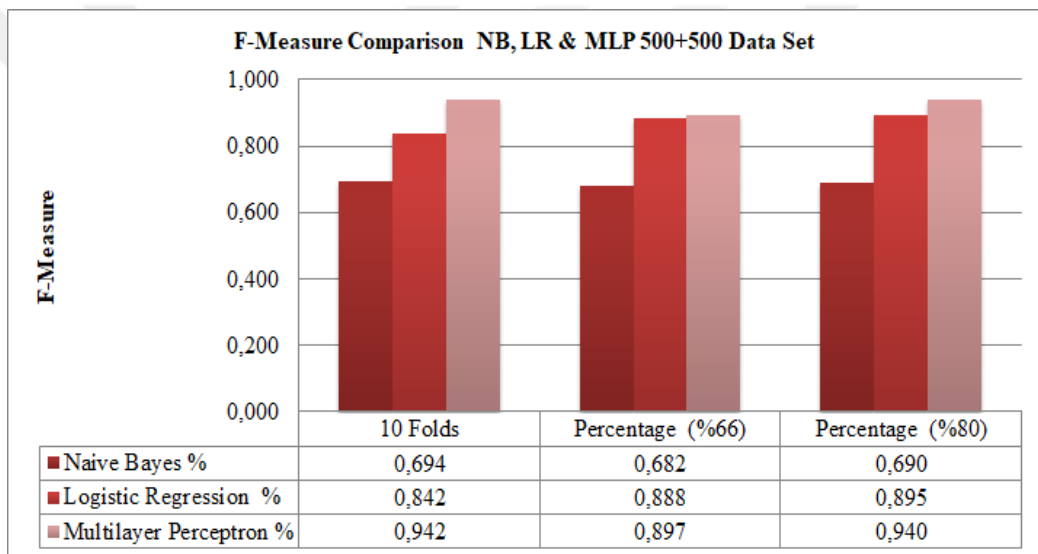


Figure 25 - F Measure Graphics for Word2Vec 500 + 500 data sets

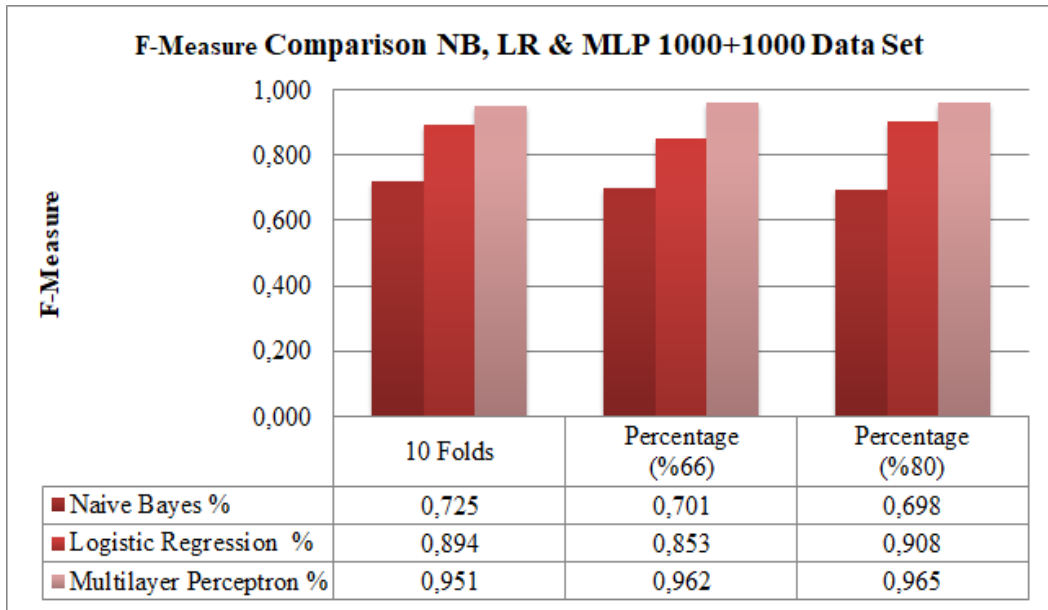


Figure 26- F Measure Graphics for Word2Vec 1000 + 1000 data sets

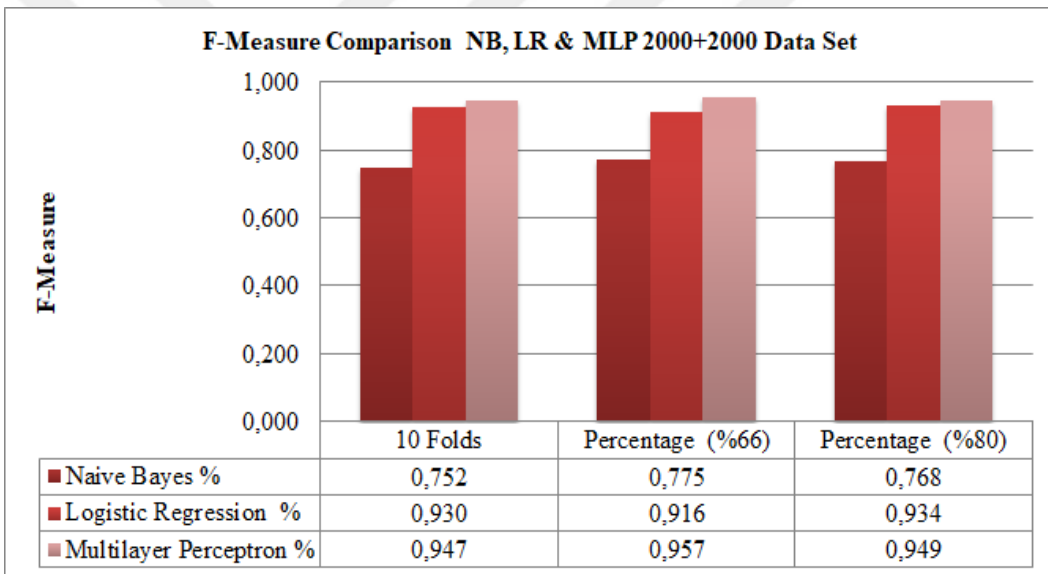


Figure 27- F Measure Graphics for Word2Vec 2000 + 2000 data sets

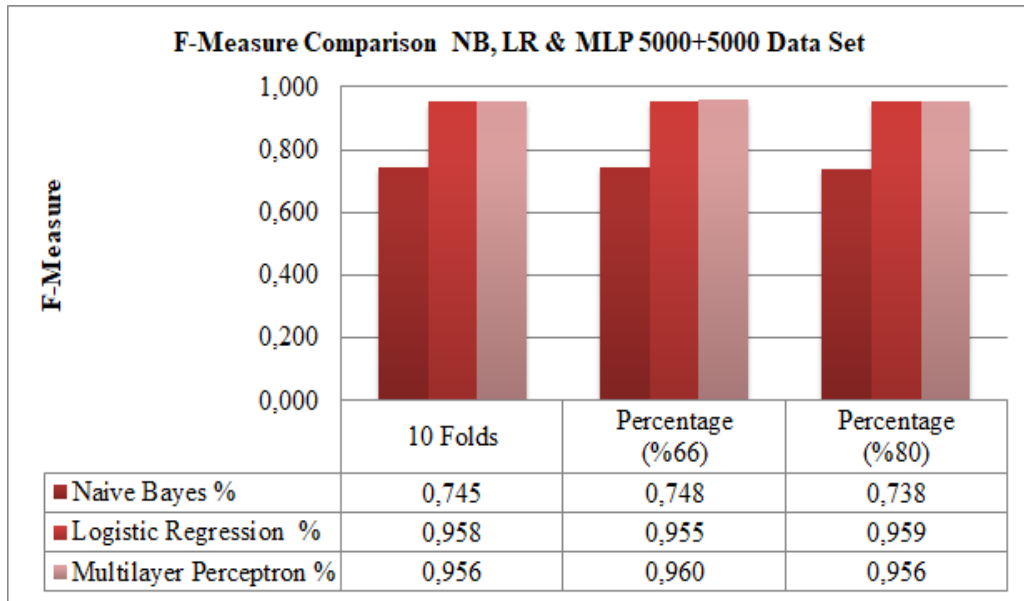


Figure 28- F Measure Graphics for Word2Vec 5000 + 5000 data sets

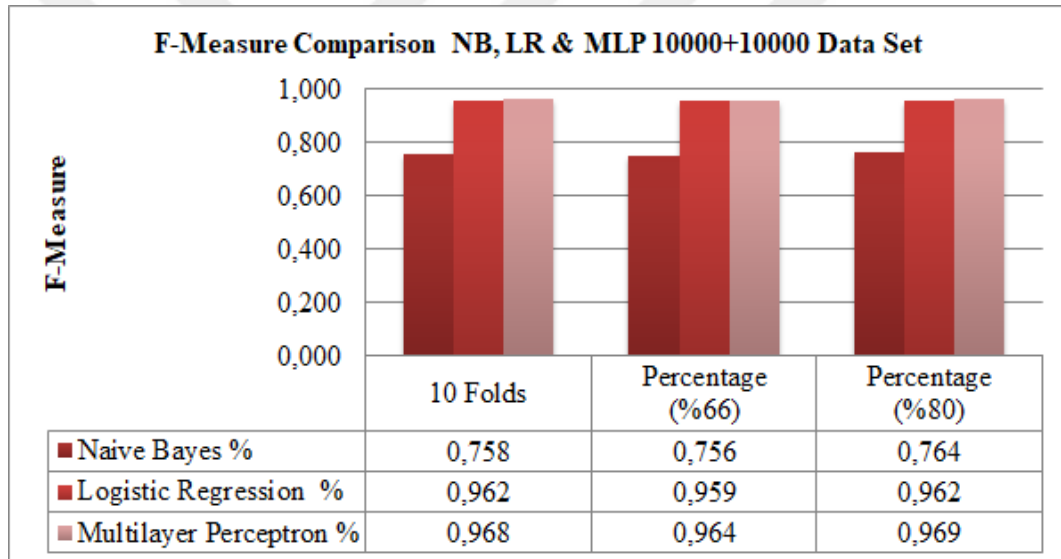


Figure 29- F Measure Graphics for Word2Vec 10000 + 10000 data sets

4.3.2.4.2 TensorFlow

By using TensorFlow with Keras, we also obtained the following results for 10-Fold Cross Validation in all datasets. MLP with Adam optimizer and 10 neuron hidden layer are used in the tests. Clearly as the dataset gets larger the accuracy goes up to 97.37% (Table 14 and Figure 30).

Table 14-TensorFlow Results for all data sets with Word2Vec

Data Sets	Cross Validation 10-Fold
300+300	94.17
500+500	94.30
1000+1000	95.40
2000+2000	95.83
5000+5000	96.80
10000+10000	97.37

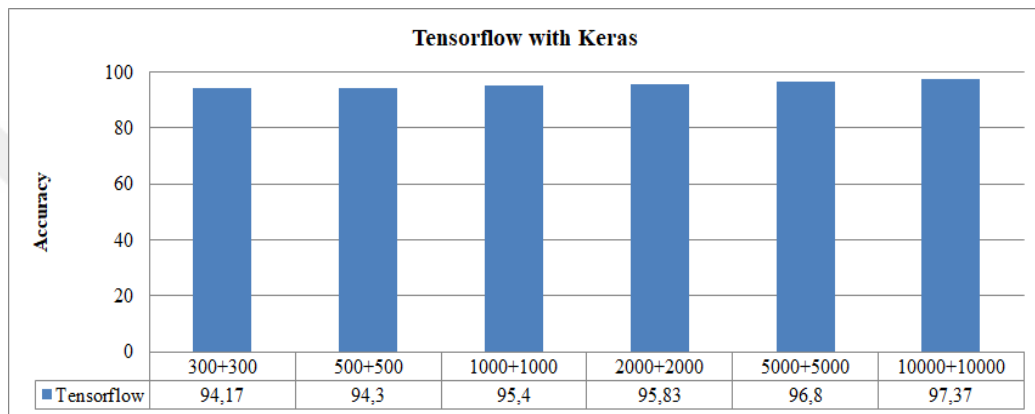


Figure 30- Accuracy Results for Tensorflow with Word2Vec

CHAPTER 5

CONCLUSION

In this thesis, we used three different methods to generate word vectors as input for machine learning algorithms. These vector representations are Weighted TF-IDF, TF-IDF using SciKit Learn, and Word2Vec. After generation of word vectors, we applied machine learning algorithms Naive Bayes (NB), SVM, Logistic Regression (LR), and deep learning algorithms, based on Multilayer Perceptron (MLP). Vector representation makes a big difference in the performance of ML algorithms. Our results show that Word2Vec data representation and MLP deep learning algorithm performs the best with big data on spam detection.

For future work, more tests with deep learning architectures (RNN, CNN, DBN) should be conducted. Vector representation using Word2Vec with better and larger training corpuses should also be done.

REFERENCES

1. **Zareapoor, M., & Seeja, K. R. (2015).** Feature extraction or feature selection for text classification: A case study on phishing email detection. *International Journal of Information Engineering and Electronic Business*, 7(2), 60.
2. **MAAWG. Messaging anti-abuse working group.** Email metrics report. Third & fourth quarter 2006. Available at [http://www.maawg.org/about/MAAWGMetric 2006 3 4 report.pdf](http://www.maawg.org/about/MAAWGMetric%2006%203%204%20report.pdf) Accessed: 04.06.07, 2006
3. **Morimoto, M., & Chang, S. (2006).** Consumers' attitudes toward unsolicited commercial e-mail and postal direct mail marketing methods: intrusiveness, perceived loss of control, and irritation. *Journal of Interactive Advertising*, 7(1), 1-11.
4. **History of WWW, W3 Organization.** <https://www.w3.org/History.html>
5. **Marmura, S. (2008).** A net advantage? The internet, grassroots activism and American Middle-Eastern policy. *New Media & Society*, 10(2), 247-271.
6. **Saab, S. A., Mitri, N., & Awad, M. (2014, April).** Ham or Spam? A comparative study for some Content-based Classification Algorithms for Email Filtering. In *Electrotechnical Conference (MELECON), 2014 17th IEEE Mediterranean* (pp. 339-343). IEEE.
7. **E-mail, Wikipedia.** <https://en.wikipedia.org/wiki/Email>.
8. **Sebastiani, F. (2002).** Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1), 1-47.
9. **Myers, K., Kearns, M., Singh, S., & Walker, M. A. (2000).** A boosting approach to topic spotting on sub dialogues. *Family Life*, 27(3), 1.

10. **Sable, C. L., & Hatzivassiloglou, V. (2000).** Text-based approaches for non-topical image categorization. *International Journal on Digital Libraries*, 3(3), 261-275.
11. **Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016).** *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.
12. **Snoek, J., Larochelle, H., & Adams, R. P. (2012).** Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems* (pp. 2951-2959).
13. **Hu, Y. T., & Schwing, A. G. (2017).** An Elevator Pitch on Deep Learning. *Get Mobile: Mobile Computing and Communications*, 21(1), 14-18.
14. **Why are deep neural networks hard to train? Neural Networks and Deep Learning.** <http://neuralnetworksanddeeplearning.com>
15. **Harris, S. R., & Gerich, E. (1996).** Retiring the NSFNET backbone service: Chronicling the end of an era. *ConneXions*, 10(4), 2-11.
16. **Reaction to the DEC Spam of 1978, Brad Templetons.** <http://www.templetons.com/brad/spamreact.html>
17. **Mail Abuse Prevention System (MAPS).** https://en.wikipedia.org/wiki/Mail_Abuse_Prevention_System
18. **Castillo, C., Donato, D., Gionis, A., Murdock, V., & Silvestri, F. (2007, July).** Know your neighbors: Web spam detection using the web topology. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 423-430). ACM.
19. **Weiss, S. M., & Indurkha, N. (1995).** Rule-based machine learning methods for functional prediction. *Journal of Artificial Intelligence Research*, 3, 383-403.
20. **Smadi, S., Aslam, N., Zhang, L., Alasem, R., & Hossain, M. A. (2015, December).** Detection of phishing emails using data mining algorithms. In *Software, Knowledge, Information Management and Applications (SKIMA), 2015 9th International Conference on* (pp. 1-8). IEEE.

21. **Shams, R., & Mercer, R. E. (2016).** Supervised classification of spam emails with natural language stylometry. *Neural Computing and Applications*, 27(8), 2315-2331.
22. **Sen, D., Das, C., & Chakraborty, S. A New Machine Learning based Approach for Text Spam Filtering Technique, (2017).**
23. **LeCun, Y., Bengio, Y., & Hinton, G. (2015).** Deep learning. *nature*, 521(7553), 436.
24. **About, Twitter.** <https://about.twitter.com/>
25. **Benevenuto, F., Magno, G., Rodrigues, T., & Almeida, V. (2010, July).** Detecting spammers on twitter. In *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)* (Vol. 6, No. 2010, p. 12).
26. **Wu, T., Wen, S., Liu, S., Zhang, J., Xiang, Y., Alrubaian, M., & Hassan, M. M. (2017).** Detecting spamming activities in twitter based on deep-learning technique. *Concurrency and Computation: Practice and Experience*, 29(19).
27. **Tang, Y. (2013).** Deep learning using linear support vector machines. arXiv preprint arXiv:1306.0239.
28. **Moradpoor, N., Clavie, B., & Buchanan, B. (2017, July).** Employing machine learning techniques for detection and classification of phishing emails. In *Computing Conference, 2017* (pp. 149-156). IEEE.
29. **Zaremba, W., Sutskever, I., & Vinyals, O. (2014).** Recurrent neural network regularization. arXiv preprint arXiv:1409.2329.
30. **Tang, J., Deng, C., & Huang, G. B. (2016).** Extreme learning machine for multilayer perceptron. *IEEE transactions on neural networks and learning systems*, 27(4), 809-821.
31. **Glorot, X., Bordes, A., & Bengio, Y. (2011).** Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 513-520).

32. **Erhan, D., Bengio, Y., Courville, A., Manzagol, P. A., Vincent, P., & Bengio, S. (2010).** Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb), 625-660.
33. **LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998).** Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.
34. **Loosli, G., Canu, S., & Bottou, L. (2007).** Training invariant support vector machines using selective sampling. *Large scale kernel machines*, 301-320.
35. **Ramos, J. (2003, December).** Using TF-IDF to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning* (Vol. 242, pp. 133-142).
36. **SciKit Learn, Machine Learning in Python.** <http://scikit-learn.org/>
37. **SciKit Learn, GitHub.** <https://github.com/scikit-learn/scikit-learn>
38. **Word2Vec, Google Code Archive.**
<https://code.google.com/archive/p/word2vec/>
39. **Word2Vec, Deep Learning for Java.**
<https://deeplearning4j.org/word2vec.html>
40. **About TensorFlow, TensorFlow.** <https://www.tensorflow.org/>
41. **Homepage, Keras.** <https://keras.io/>
42. **Naive Bayes, Web Page of Dr. Saed Sayad.**
http://www.saedsayad.com/naive_bayesian.htm
43. **Ruck, D. W., Rogers, S. K., & Kabrisky, M. (1990).** Feature selection using a multilayer perceptron. *Journal of Neural Network Computing*, 2(2), 40-48.
44. **Kleinbaum, D. G., & Klein, M. (2010).** Introduction to logistic regression. In *Logistic regression* (pp. 1-39). Springer, New York, NY.
45. **Press, S. J., & Wilson, S. (1978).** Choosing between logistic regression and discriminant analysis. *Journal of the American Statistical Association*, 73(364), 699-705.
46. **Here is What You Get, Comodemia.** <https://comodemia.comodo.com/>

47. **Davis, J., Hossain, L., & Murshed, S. H. (2007).** Social network analysis and organizational disintegration: the case of Enron corporation. ICIS 2007 Proceedings, 5.
48. **Chandrasekaran, M., Narayanan, K., & Upadhyaya, S. (2006, June).** Phishing email detection based on structural properties. In NYS Cyber Security Conference (Vol. 3).
49. **Parsons, K., McCormac, A., Pattinson, M., Butavicius, M., & Jerram, C. (2015).** The design of phishing studies: Challenges for researchers. *Computers & Security*, 52, 194-206.
50. **Adewumi, O. A., & Akinyelu, A. A. (2016).** A hybrid firefly and support vector machine classifier for phishing email detection. *Kybernetes*, 45(6), 977-994.
51. **Moghimi, M., & Varjani, A. Y. (2016).** New rule-based phishing detection method. *Expert systems with applications*, 53, 231-242.
52. **Diale, M., Van Der Walt, C., Celik, T., & Modupe, A. (2016, November).** Feature selection and support vector machine hyper-parameter optimization for spam detection. In Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech), 2016(pp. 1-7). IEEE.
53. **Agarwal, D. K., & Kumar, R. (2016).** Spam Filtering using SVM with different Kernel Functions. *International Journal of Computer Applications*, 136(5).
54. **Jain, G (2016).** A Study of Bayesian Classifiers Detecting Gratuitous Email Spamming. *Communications on Applied Electronics*6(2):26-30.
55. **Hashemi, S. M, (2015).** Detection and Filtering Spam using Feature Selection and Learning Machine Methods. *Journal of Academic and Applied Studies (Special Issue on Engineering & Applied Sciences)* Vol. 5(4) April 2015, pp. 14-30
56. **Al Sarhan, A., Jabri, R., & Sharieh, A. (2017).** Website Phishing Detection Using Dom-Tree Structure and Cant-MinerPB Algorithm. *American Journal of Computer Science and Information Engineering*, 4(4), 38-42.
57. **Altaher, A. (2017).** Phishing websites classification using hybrid svm and knn approach. *Int J Adv Comput Sc*, 421, 8.
58. **Abu-Nimeh, S., Nappa, D., Wang, X., & Nair, S. (2007, October).** A comparison of machine learning techniques for phishing detection. In Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit (pp. 60-69). ACM.
59. **Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C. D., & Stamatopoulos, P. (2000).** Learning to filter spam e-

mail: A comparison of a naive Bayesian and a memory-based approach. arXiv preprint cs/0009009.

60. **Awad, W. A., & ELseuofi, S. M. (2011).** Machine Learning methods for E-mail Classification. *International Journal of Computer Applications*, 16(1).
61. **Lai, C. C., & Tsai, M. C. (2004, December).** An empirical performance comparison of machine learning methods for spam e-mail categorization. In *Hybrid Intelligent Systems, 2004. HIS'04. Fourth International Conference on* (pp. 44-48). IEEE.
62. **Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013).** Applied logistic regression (Vol. 398). John Wiley & Sons.
63. **Wu, C. H. (2009).** Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks. *Expert Systems with Applications*, 36(3), 4321-4330.
64. **Gray, A., & Haahr, M. (2004, July).** Personalized, Collaborative Spam Filtering. In CEAS.

APPENDICES A

CURRICULUM VITAE



PERSONAL INFORMATION

Surname, Name: Onur Göker

Nationality: Turkish (TC)

Date and Place of Birth: 14/04/1989

Marital Status: Single

Phone: +90 555 299 29 00

Email: onurgoker89@gmail.com

EDUCATION

Degree	Institution	Year of Graduation
MSc	Çankaya Univ., Computer Engineering	2014-2018
B.Sc.	TOBB ETU, Computer Engineering	2006-2011

WORK EXPERIENCE

Year	Place	Enrollment
2017-	Comodo Inc.	Software Team Lead
2015-2017	Comodo Inc.	Senior Software Developer
2014-2015	Sentim Bilişim A.Ş.	PHP Software Dev.
2013-2014	Genel Bilgi Teknolojileri	Web Developer
2013-2013	Positive - A Digital Approach	Web Developer
2011-2013	Beril Teknoloji	Web Developer
2010	Kasırga Bilişim Ltd. Şti.	Intern
2008	Library, TOBB ETÜ	Intern
2007	IT Centers, TOBB ETÜ	Intern

FOREIGN LANGUAGES

Turkish	: Native
English	: Professional
German	: Elementary
Portuguese	: Beginner

PROJECTS

1. Comodo One Remote IT Management Tools - Service Desk
2. Comodo One Remote IT Management Tools - CRM
3. Comodo One Remote IT Management Tools - ITSM
4. Ministry of Health's Prescription Information System
5. Hisarlar Corporate Website
6. Tepe Savunma Corporate Website
7. Turkar Corporate Website
8. Toyzzshop Corporate Web Site
9. Petlas Corporate Website
10. Starmaxx Corporate Website
11. Kürk İnşaat Corporate Web Site

12. Üzümcü Corporate Website
13. Vamos Sports Complex Corporate Website
14. Şirketçe Classified Advertising Project
15. Tulpar JAVA Compiler
16. Uygulamam.com Mobile Application Project

CERTIFICATE

Bilge Adam MCSD Program

C#, .Net Framework, SQL, ASP.Net, Azure, Windows Phone technologies

PUBLICATIONS

1. **Goker, O., Nazli, N., Dogdu, E., Choupani R., Erol, M.M., (2018).** A Robust Watermarking Scheme Over Quadrant Medical Image in Discrete Wavelet Transform Domain. International Conference on Control, Decision and Information Technologies 2018 (CODIT'18).
2. **Hassanpour R., Dogdu E, Choupani R, Goker O, Nazli N, (2018).** Phishing E-mail Detection by Using Deep Learning Algorithms. Poster, supplemental material(s). ACM SE '18: Southeast Conference Proceedings.