

**IMMUNE-INSPIRED PREDICTIVE MODELS FOR
EMAIL CLASSIFICATION AND SPAM FILTERING**

BY

ALI A. AL-HASAN

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER SCIENCE


December 2015

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS


DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

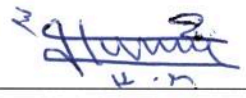
This thesis, written by **ALI A. AL-HASAN** under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER SCIENCE**.




Dr. El-Sayed M. El-Alfy 22/12/15
(Thesis Advisor)



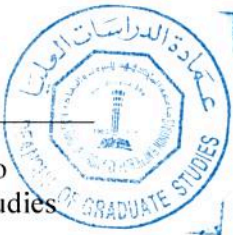
Dr. Abdulaziz M. Al-Khoraidly
ICS Department Chairman

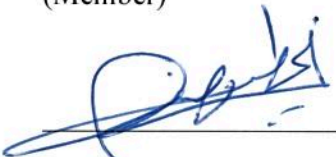


Dr. Husni A. Al-Muhtaseb 24 Dec 2015
(Member)



Dr. Salam A. Zummo
Dean of Graduate Studies





Dr. Wasfi G. Al-Khatib
(Member)

28/12/15

Date

DEDICATION

I dedicate this work to my parents for their endless support and inspiration.

ACKNOWLEDGMENTS

Acknowledgment is due to God (Allah) for giving me the health, strength, knowledge and everything to complete this research.

Special thanks to Dr. El-Sayed M. El-Alfy for his patience, guidance, support and generous contribution of his time and expertise to achieve this work and making it possible. He dedicated a lot of time in advising, reviewing and commenting on this work. I really appreciate his confidence in me and he was never busy to set, explain and work with me. He graciously accepted many late night hours in assembling and completing this work. He inspired me, set me on the right track and gave me a push in the right direction to complete this work. I would also like to thank the committee members, Dr. Husni Al-Muhtaseb and Dr. Wasfi Al-Khatib, for their continuous support from the initial idea and proposal up to the printing stage of this thesis.

I would like also to express my appreciation to everyone who encouraged and supported me including my parents, brothers, family and friends.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	IV
TABLE OF CONTENTS	V
LIST OF TABLES.....	VIII
LIST OF FIGURES	X
LIST OF ABBREVIATIONS	XII
ABSTRACT	XIII
ملخص الرسالة	XIV
CHAPTER 1 INTRODUCTION.....	1
1.1 Email Basics	5
1.2 Thesis Objectives.....	7
1.3 Motivation.....	7
1.4 Thesis Organization.....	9
CHAPTER 2 BACKGROUND AND LITERATURE REVIEW	10
2.1 Email Classification and Spam Filtering.....	10
2.1.1 Datasets	11
2.1.2 Feature Extraction	20
2.1.3 Feature Selection Methods.....	23
2.1.4 Filter Feature Selection	24
2.1.5 Wrapper Feature Selection	27
2.1.6 Embedded Feature Selection	28

2.2	Machine Learning Algorithms.....	29
2.2.1	Support Vector Machine (SVM)	29
2.2.2	Bayesian Learning.....	30
2.2.3	Lazy Learning	31
2.2.4	Rule Based	31
2.2.5	Others	32
2.3	Biological Background.....	36
2.3.1	Innate Immune System	39
2.3.2	Adaptive Immune System	40
2.4	AIS Algorithms and Applications	41
2.4.1	Negative Selection Algorithm (NSA).....	42
2.4.2	Clonal Selection Algorithm (CSA)	43
2.4.3	Idiotypic Network Based Algorithms.....	44
2.4.4	AIS Applications.....	44
2.5	Biological & Artificial Dendritic Cells	45
2.5.1	Classical Immune System Theory.....	46
2.5.2	Danger Theory	47
2.5.3	Dendritic Cell Algorithm (DCA).....	50
2.5.4	DCA Related Work.....	55
CHAPTER 3 METHODOLOGY.....		56
3.1	Proposed Model Overview	56
3.2	Feature Extraction.....	58
3.2.1	Spam Email Feature Extraction:.....	59
3.2.2	Email Multi-Category Classification Feature Extraction:.....	62
3.3	Feature Selection	64

3.4	Signals Generator	64
3.5	Fusion Using DCA	66
CHAPTER 4 EXPERIMENTS.....		69
4.1	Experiments Setup	69
4.2	Datasets.....	70
4.3	Evaluation Measures.....	71
4.4	Spambase and SpamAssassin Experiment (using 3C-DCA)	74
4.5	Spam Experiments (using 2c-DCA).....	80
4.5.1	SpamAssassin dataset	84
4.5.2	TREC dataset.....	86
4.5.3	SMS Big dataset:.....	89
4.5.4	SMS UCI dataset:	91
4.6	Email Message Multi-Category Classification	94
CHAPTER 5 CONCLUSION AND FUTURE WORK.....		97
5.1	Conclusion	97
5.2	Future work.....	99
REFERENCES		100
VITAE		112

LIST OF TABLES

Table 1: Worldwide Daily Email Traffic per year [1]	2
Table 2: Preprocessed spam email datasets	17
Table 3: Raw spam email datasets	18
Table 4: Spam SMS datasets	19
Table 5: Bag of word vector creation.....	21
Table 6: Spam feature extraction	22
Table 7: Features weights	27
Table 8: Comparison of classifiers.....	35
Table 9: Examples of function words	62
Table 10: Sample of stop words.....	63
Table 11: Examples of rooted words using Porter Algorithm	63
Table 12 Machine Learning Configuration	70
Table 13: Used Datasets	70
Table 14: Confusion matrix	71
Table 15 DCA default parameters value.....	75
Table 16: DCA AUC performance by changing Antigen multiplier and DCs	75
Table 17: DCA performance for several signals weights.....	76
Table 18: Best DCA parameters values	77
Table 19: Performance for several classifiers on Spambase dataset.....	78
Table 20: Performance for several classifiers on SpamAssassin dataset.....	79
Table 21: Gini index weight SMS Big.....	81
Table 22: Gini index weight SMS UCI	81
Table 23: Gini index weight SpamAssassin	82
Table 24: Gini index weight TREC2005	82
Table 25: NB Classifier	83
Table 26: SVM Classifier	83
Table 27: KNN Classifier	83
Table 28: DCA AUC performance for SpamAssassin	84
Table 29: DCA performance for several signals weight for SpamAssassin dataset.....	85
Table 30: Best DCA parameters value for SpamAssassin dataset	85
Table 31: Performance for several classifiers for SpamAssassin dataset.....	85
Table 32: DCA AUC performance TREC dataset	87
Table 33: DCA performance for several signals weight for TREC dataset	87
Table 34: Best DCA parameters value for TREC dataset	88
Table 35: Performance for several classifiers for TREC dataset	88
Table 36: DCA AUC performance for SMS Big dataset.....	89
Table 37: DCA performance for several signals weight for SMS Big dataset.....	90
Table 38: Best DCA parameters value for SMS Big dataset.....	90
Table 39: Performance for several classifiers for SMS Big dataset.....	90

Table 40: DCA AUC performance for SMS UCI dataset	92
Table 41: DCA performance for several signals weight for SMS UCI dataset	92
Table 42: Best DCA parameters value for SMS UCI dataset	93
Table 43: Performance for several classifiers for SMS UCI dataset	93
Table 44: Users datasets details.....	95
Table 45: Comparison of percentage classification accuracy per user	96

LIST OF FIGURES

Figure 1: Global spam and email volume (trillions of messages per quarter) [3].....	3
Figure 2: Message structure and related features	5
Figure 3: Standard email message process	5
Figure 4: Example of email header (private information is replaced with X's).....	6
Figure 5: Attack Sophistication vs. Intruder Technical Knowledge	8
Figure 6: General layout of anti-spam system.....	11
Figure 7: Enron email directory structure.....	14
Figure 8: Feature Selection Process	24
Figure 9: Filter feature selection model	25
Figure 10: The wrapper approach for feature selection	28
Figure 11: Forward feature selection algorithm	28
Figure 12: SVM binary classification (a) small margin, (b) large margin, (c) non-linear	30
Figure 13: Artificial immune system as a branch of computational intelligence.....	36
Figure 14: The evolution of AIS [56].	37
Figure 15: Human immune system locations	38
Figure 16: Human immune system layers	38
Figure 17: Innate and adaptive immune system cells	39
Figure 18: Adaptive immune system response time	40
Figure 19: Non-self vs. self-pattern	43
Figure 20: Antigens carry marker molecules that identify them as foreign	46
Figure 21: Danger theory model	47
Figure 22: Transformation between DC states [60].....	49
Figure 23: DC states [60].....	49
Figure 24: A depiction of the process used to develop the DCA	51
Figure 25: Signals weights and transformation to output	52
Figure 26: DCA three stages [61]	53
Figure 27: DCA pseudo code	54
Figure 28: Decision of one cell	54
Figure 29: Layout of the proposed system	57
Figure 30: Feature extraction for spam filtering	57
Figure 31: Feature extraction for email classification	58
Figure 32: Example of tagging for organizations.....	62
Figure 33: Process for calculating PAMP, safe and danger signals.....	66
Figure 34: Pseudo code for DCA	68
Figure 35: Sample of top 150 antigen MCAV values.....	77
Figure 36: ROC performance comparison of various classifiers on Spambase dataset	79
Figure 37: ROC performance comparison of various classifiers on SpamAssassin	80
Figure 38: ROC for several classifiers for SpamAssassin dataset	86
Figure 39: ROC for several classifiers for TREC dataset.....	88

Figure 40: ROC for several classifiers for SMS Big dataset	91
Figure 41: ROC for several classifiers for SMS UCI dataset	93
Figure 42: One-versus-all	94
Figure 43: Comparison of classification accuracy per user	96

LIST OF ABBREVIATIONS

Acc	Accuracy
AUC	Area Under Curve
BoW	Bag of Words
DCA	Dendritic Cell Algorithm
FM	F-Measure
FP	False Positive
GI	Gini Index
KNN	K-Nearest Neighbor
MCAV	Mature Context Antigen Value
NB	Naïve Bayes
PAMP	Pathogen-Associated Molecular Pattern
ROC	Receiver Operating Characteristic
SMS	Short Message Service
SP	Spam Precision
SR	Spam Recall
SVM	Support Vector Machine
TP	True Positive
VSM	Vector Space Model

ABSTRACT

Full Name : Ali A. Al-Hasan
Thesis Title : IMMUNE-INSPIRED PREDICTIVE MODELS FOR EMAIL
CLASSIFICATION AND SPAM FILTERING
Major Field : Computer Science
Date of Degree : December, 2015

Electronic messages have become the most popular, frequently-used and powerful medium for communications. However, everyone receives thousands of messages that require manual sorting into different folders. Moreover, one of the major issues and annoying problems in information security domain faced by end-users is receiving a large volume of unwanted messages, also known as spam. Nowadays, most messaging systems have built-in filtering mechanisms that can block or quarantine unwanted messages based on predefined keywords. Over the years, some extensions for these filters have been proposed for improving their performance. However, spamming techniques have also continued to evolve and bypass existing countermeasures. Hence, new solutions and ideas must be explored. This thesis discusses the electronic mail classification and spam filtering problems. It reviews related issues and their impact. It also analyzes techniques used by spammers, and evaluates and compares the most common machine learning paradigms to classify emails and distinguish spam messages. Moreover, in this thesis, we explore a novel and promising methodology inspired by the danger theory of the human immune system to design hybrid approaches for constructing email classifiers and spam filters. Additionally, we study a number of ways to extract and select more relevant features to reduce the complexity and improve the performance. The proposed method is evaluated on a number of benchmark datasets. The results demonstrate that the proposed method is a promising solution for textual email classification and spam filtering.

ملخص الرسالة

الاسم الكامل: علي عبدالرزاق الحسن

عنوان الرسالة: نماذج تنبؤ مستوحاة من نظام المناعة لتصنيف رسائل البريد الإلكتروني وترشيح الرسائل غير المرغوب فيها

التخصص: علوم الحاسب

تاريخ الدرجة العلمية: ديسمبر 2015

تعد الرسائل الإلكترونية من الوسائل المنتشرة والفعالة للتواصل في وقتنا الحالي على المستوى الشخصي والرسمي. ويتلقى المستخدمون كثيراً من الرسائل التي تتطلب فرزاً يدوياً لمجلدات مختلفة، كما يتلقون كمّاً هائلاً من الرسائل التي لا طائل منها أو غير المرغوب فيها والتي تعرف بالبريد المزعج (SPAM). وتعد هذه القضية من القضايا الأمنية المهمة التي تتطلب عناية خاصة من المستخدمين ومديري الشبكات. ولذا تحتوي معظم أنظمة الرسائل الإلكترونية على آليات للحد من هذه المشكلة. إلا أنه يلزم استحداث طرق لتوائم التطور في الأنظمة الحديثة.

تقوم هذه الدراسة بتحليل فعالية الحلول المتاحة لتصنيف الرسائل الإلكترونية النصية وترشيح غير المرغوب منها. كما تقترح طرقاً لاستخلاص سمات أكثر فعالية للتمييز بين الرسائل المختلفة. وعلاوة على ذلك، تقوم ببناء نماذج تصنيف مبنية على خوارزمية مستوحاة من نظم المناعة الطبيعية. كما تقوم بتحليل أداء النماذج المقترحة ومقارنتها بطرق أخرى باستخدام قواعد بيانات ذات خصائص مختلفة. وقد أظهرت النتائج فعالية الحل المقترح في تصنيف وترشيح الرسائل الإلكترونية النصية.

CHAPTER 1

INTRODUCTION

The Internet is playing a crucial role in our daily life. Electronic mail message service (email) is one of the earliest, most popular, and frequently used Internet applications that provides a powerful medium for personal and business communications. This is due to its flexibility, worldwide accessibility, relatively fast message transfer and low infrastructure costs. However, users receive large number of mail messages daily as shown in Table 1. For example, the number of sent and received emails in 2015 is over 205 billion. This number is expected to increase every year by 5%. The popularity of electronic email service has resulted in several security issues facing its users. Email spam is one of the major issues annoying end-users and causing various damages including financial loss to companies and law violation. Spam impacts an organizations' and individuals' network bandwidth, storage space, and system computational power. In addition, it wastes users' time and productivity by requiring them to look through and sort a large volume of email messages, which is annoying and might violate their privacy. Spam may also advertise and broadcast prohibited material such as pornography content. Moreover, it might be a vector for other crimes or illegal activities such as gambling, fraud, ransom, and intimidation.

Table 1: Worldwide Daily Email Traffic per year [1]

Daily Email Traffic		Year				
		2015	2016	2017	2018	2019
Total worldwide emails sent/received per day	Numbers in billion	205.6	215.3	225.3	235.6	246.5
	Growth	-	5%	5%	5%	5%
Business emails sent/received per day	Numbers in billion	112.5	116.4	120.4	124.5	128.8
	Growth	-	3%	3%	3%	3%
Consumer emails sent/received per day	Numbers in billion	93.1	98.9	104.9	111.1	117.7
	Growth	-	6%	6%	6%	6%

There are various definitions for spam email and how it is different from legitimate (aka ham) email. Among the existing definitions, email spam is: “*unsolicited, unwanted email that was sent indiscriminately, directly or indirectly, by a sender having no current relationship with the user*” [2]. Identifying and filtering out these types of messages based on the sender information alone is hard. When the Simple Mail Transfer Protocol (SMTP) was designed, security was not a crucial issue. The mitigation solutions for spam email involve both non-technical and technical factors. The non-technical solutions involve human training and email policies. The technical solutions depend on deploying a multitude of anti-spammers and policy enforcement systems.

According to McAfee Quarterly Threat report in 2015, the global spam volume increased every year (as shown in Figure 1) and much of the growth is driven by legitimate “affiliate” marketing firms using an untrustworthy list of brokers [3]. The risk of spam is very high as it may: waste computing resources, contain malicious attachments, market fraudulent products and services, and steal confidential information using phishing attacks techniques.

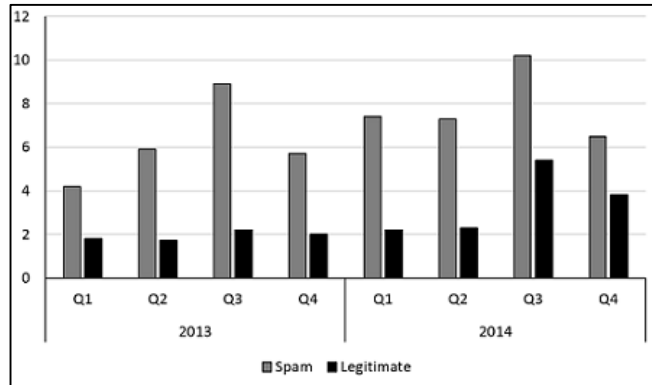


Figure 1: Global spam and email volume (trillions of messages per quarter) [3]

Most of the spam is nowadays generated by automated tools which populate the user's inbox with unsolicited junk messages; thus negatively impacts storage space, network traffic and system computational power. In addition, spam wastes users' time and productivity as they need to check unnecessarily larger number of messages in the inbox and even other messages that were quarantined by deployed filters. Spam may also advertise and broadcast prohibited material such as pornography content as well as illegal or criminal information that is related to gambling, fraud, ransom, intimidation, illegal sale of guns, fake winning and/or ammunition.

Spam email messages might contain malicious computer programs (such as viruses, Trojans, worms, or bots), which can harm the system and make it vulnerable to further attacks. For example, "ILoveYou", Nimda, Melissa, and others were spread through spam. In February 2015, the global ratio of spam in email traffic was 54% [4]. According to [5], the financial losses caused by spam in 2009 were \$130 billion. These losses were calculated based on: user productivity cost (deleting spam, looking for false positives),

helpdesk cost (IT helping end users deal with spam), spam control software, hardware and services (licensing fees, amortized capital costs, etc.).

A spam email message has some characteristics related to its content and traffic that can help in distinguishing it from a legitimate email messages. Figure 2 shows the structure of an email message and examples of content related features [6]. As shown, there are two main components: header and message body. The header is composed of a set of fields; each has a name, value and meaning. The body is mostly text in HTML format with graphical elements. These characteristics and others such as email traffic can be used to filter an email. For example, spam arrival rate is more stable over time than for legitimate emails [7]. Most spammers hide their identities which come from small concentrated part of IP address space. Due to the significant technical differences between delivery methods, the spam problem is considered complex. There are many ways to filter out spam. For example, recognizing spam from legitimate email could be achieved by analyzing the contents and the method of delivery of the messages. Other approaches would utilize user's personal judgment to define spam in addition to message content. Understanding the spamming tricks would definitely assist in developing adequate solutions to prevent and mitigate spam email. Due to this continuous evolution of the spamming techniques, systems for detecting and filtering spam have to be upgraded and new approaches need to be investigated.

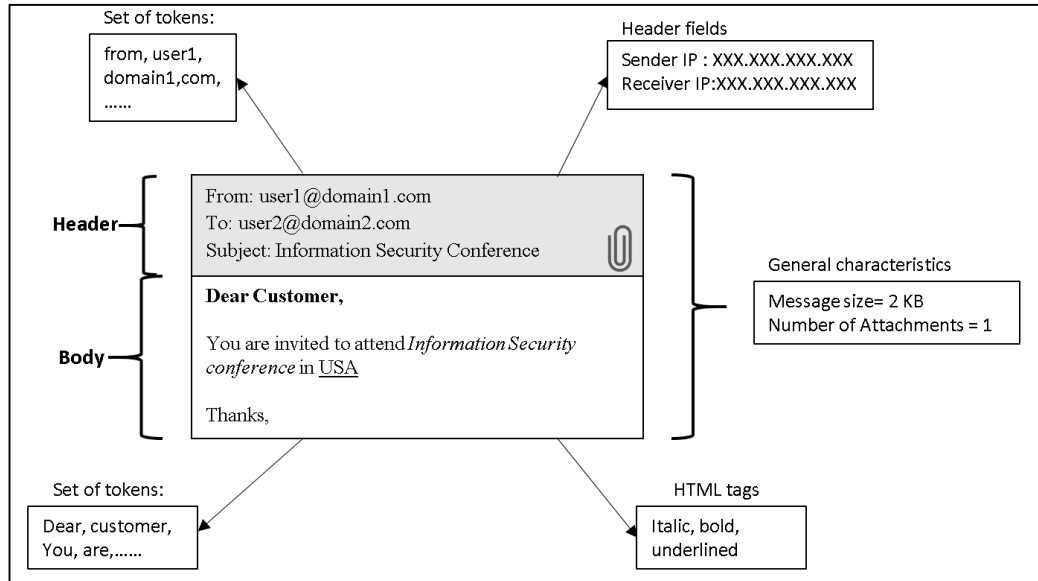


Figure 2: Message structure and related features

1.1 Email Basics

An email message contains several elements that enable it to be transferred and routed from sender's domain/inbox to recipient's domain/inbox. Figure 3 depicts how two email servers are communicating and email is routed.

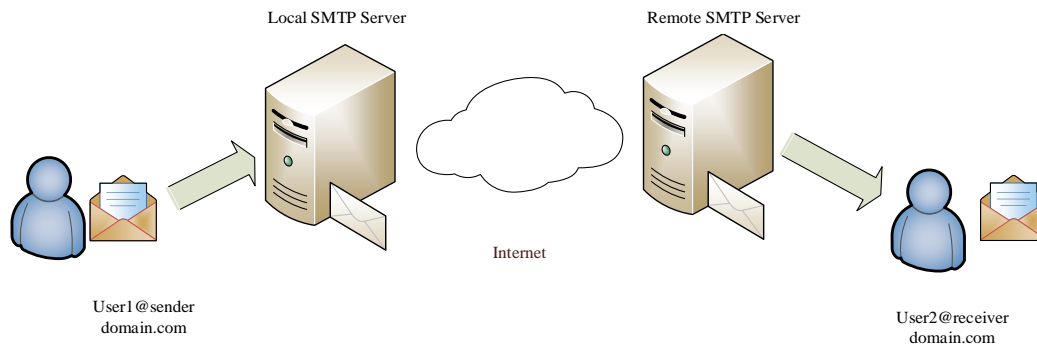


Figure 3: Standard email message process

An email message consists of two main parts: email header and email body. The email header is added into the email message automatically by the email server. The addition of the header is performed during email composition and transformation between systems [8]. It contains valuable information about the sender, mail system, and path taken to deliver the message. An illustrative example is shown in Figure 4. On the other hand, the body of the email message contains text and/or images as composed by the sender.

```
Delivered-To: aXXXXXXX@gmail.com
Received: by 10.XX.XX.XX with SMTP id rXXcspXXX153XXXa;
Sun, XX Mar 201X 19:33:22 -0700 (PDT)
X-Received: by 10.XX.XX.XX with SMTP id g30mr62355933qkh.54.1427682800909;
Sun, XX Mar 201X 19:33:20 -0700 (PDT)
Return-Path: <XXXXXXX@XXXXXXX.com>
Received: from smtpX.XXXXXXXX.com (smtpX.XXXXXXXX.com. [174.XX.XX.XX])
by mx.google.com with ESMTP id 17XXXXXX388XXXXXX1.47.201X.0X.XX.XX.XX.XX
for <aXXXXXXX@gmail.com>;
Sun, XX Mar 201X 19:33:20 -0700 (PDT)
Received-SPF: pass (google.com: domain of XXXXXXXXXX@XXXXXXX.com designates
174.XX.XX.XX as permitted sender) client-ip=174.XX.XX.XX;
Authentication-Results: mx.google.com;
spf=pass (google.com: domain of XXXXXXXXXX@XXXXXXX.com designates 174.XX.XX.XX as
permitted sender) smtp.mail=XXXXXX@XXXXXXXXXXXX.com;
dkim=pass (test mode) header.XX=@XXXXXXX.com;
dmarc=pass (p=NONE dis=NONE) header.from=XXXXXXX.com
```

Figure 4: Example of email header (private information is replaced with X's)

1.2 Thesis Objectives

The main objectives of this research work are to:

- Study the state-of-the-art techniques for text spam filtering including feature extraction and selection, and classification algorithms.
- Gain understanding of the computational algorithms inspired by the immune-system with focus on the Dendritic Cell Algorithm (DCA).
- Investigate and develop new models based on DCA to classify text emails and filter out spam messages.
- Evaluate the effectiveness of the proposed models and their integration with other machine learning techniques and compare their performance with other approaches in the literature.
- Study a number of ways to extract and select the most relevant features to reduce the complexity and enhance the performance.

In this work, we consider only textual email messages. We have not considered embedded images or email traffic.

1.3 Motivation

As mentioned in the introduction section, spam messages are not only annoying and waste a lot of resources but they can also be a medium to distribute malicious codes and undermine the information security systems of an organization. With the improvement of

information security controls and measures, spammers have also improved their techniques in order to wage successful attacks. As shown in Figure 5, the required knowledge for hackers is getting steadily lower, yet they can launch more sophisticated attacks. Similar issue is facing anti-spam solutions. Although different spam filters and approaches have been developed, spammers are still finding ways to bypass them. This motivates the need to explore and develop new methodologies for adequate spam filtering. One of the approaches that have been recently proposed based on the danger theory of biological immune system is known as Dendritic Cell Algorithm (DCA). This approach has potential and natural characteristics that sound promising for mitigating the risk of spam. Moreover, it does not require high computing resources. Therefore, in this thesis we will design and evaluate a novel model based on DCA for email classification and filtering.

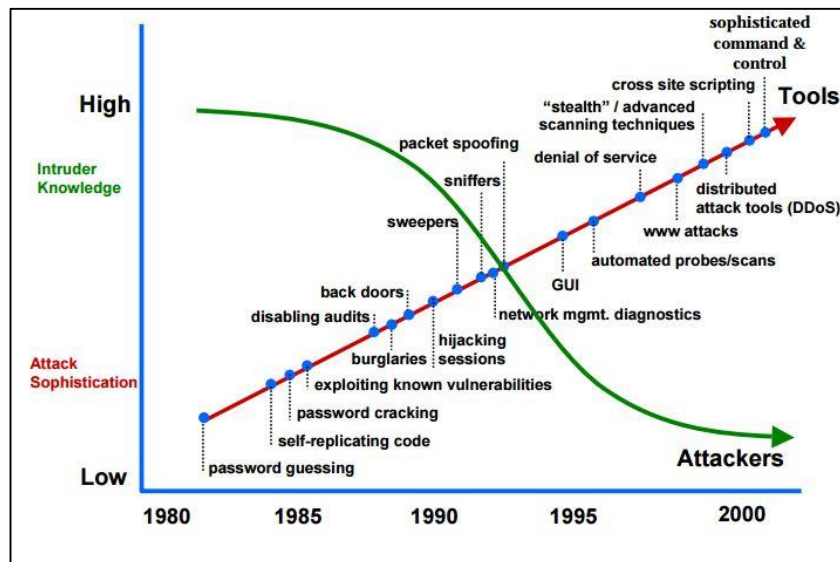


Figure 5: Attack Sophistication vs. Intruder Technical Knowledge¹

¹ <http://www.sei.cmu.edu/reports/02sr009.pdf>

1.4 Thesis Organization

In addition to this chapter, this thesis consists of four more chapters. In Chapter 2, we discuss literature review and background research about text mining and classification in general and spam filtering in particular including feature extraction, feature selection and classification algorithms. In addition, biological and artificial immune systems are explained with more focus on the biological dendritic cells and the dendritic cell algorithm. Chapter 3 explains our proposed methodology and architecture for message classification and spam filtering. In chapter 4, experimental works and used procedures are described in addition to giving an overview of the used datasets and evaluation measures. Chapter 5 summarizes the thesis contributions and findings, and provides recommendation for future work.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

2.1 Email Classification and Spam Filtering

Document classification problem has been widely researched in databases, data mining, and information retrieval. This problem is defined in general as follows. There are a set of training records $T = \{X_1, X_2, \dots, X_N\}$, such that each record is labeled with a class value $\{1, 2, \dots, k\}$, where $k \geq 2$. It is required to construct a computational model that can be used to predict the class label for a given test instance with unknown class label. In our work, we consider two special cases: (a) $k = 2$ (spam filtering), (b) $k > 2$ (the more general multi-category email classification problem). Figure 6 shows a typical spam filtering system. It consists of two main phases:

- 1) Training phase (also known as offline phase): in this phase the machine learning algorithm is trained on a sample of the dataset (training set). The output of this phase is a model that can be used to predict and classify unseen data. There are several processes that must be performed to generate the classifier model. The main processes are: feature extraction, feature selection and learning algorithm. More about these processes will be discussed in the next section.
- 2) Prediction phase (also called online or deployment phase): in this phase the system predicts and classifies a new email to either spam or non-spam using the model generated from the training phase. Some processes are needed similar to the processes that were used in the previous phase, i.e. feature extraction and feature selection. In this phase, user feedback can be utilized to enhance and adjust the classification model.

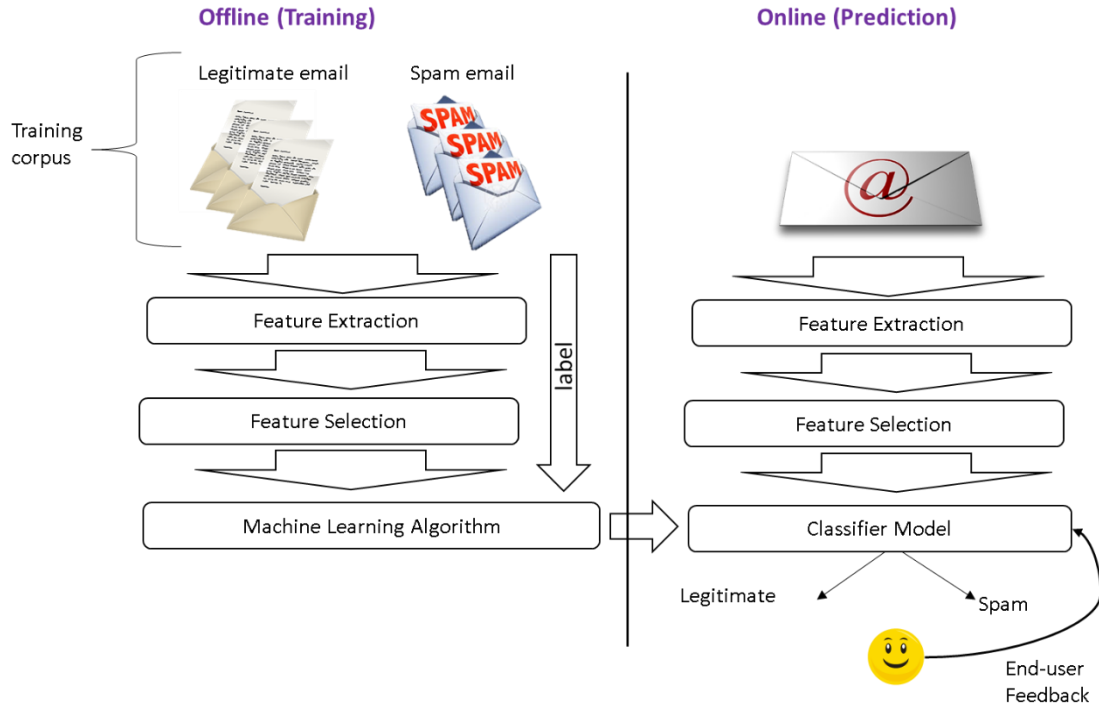


Figure 6: General layout of anti-spam system

The upcoming subsections provide an overview of datasets, spam detection approaches, feature selections and extraction methods that are fed into the classification algorithms, and immune based algorithms.

2.1.1 Datasets

Evaluating the proposed approach on several datasets will help us to benchmark our model against other widely used classifiers. Also, the datasets are used to train and build the classifiers during the online phase. However, collecting and obtaining the right datasets with enough number of messages is a challenging task. Due to privacy reasons, it is not easy to obtain non-spam (legitimate) emails. Moreover, mixing a dataset from different sources might lead to biased training of the classifier since the corpus distribution might

not reflect the real distribution. Therefore, it is recommended to obtain the collection from the same source that the filter will be trained on. Because of the privacy issue, some researchers have utilized either datasets that are not publicly accessible or a combination of private and public datasets in their research.

In the literature, a number of spam corpora are available to the public for research purpose, e.g. Spambase, SpamAssassin, and LingSpam. Each dataset has specific features and characteristics including the number of users considered, compilation time, and general subject of the messages included. These datasets are available in different formats and have been used in several researches to evaluate various spam filtering techniques. Some of them are in raw format while others are preprocessed with a limited number of attributes. To protect end-users privacy some datasets have masked some terms. Table 2, Table 3, and Table 4 show examples of the widely used corpora for electronic messages.

Spambase dataset² has been widely tested in studies of general machine learning classifiers. It is a publicly accessible dataset in UCI Machine Learning Repository. This database was created at Hewlett-Packard Labs in 1999. It has 4601 emails; 1813 (39.4%) of them are spam. It is available in a preprocessed format where each email is described by 58 attributes (57 numeric attributes and a classification label, i.e. spam or non-spam). Most of the attributes indicate the occurrence frequency of a particular word or character in the e-mail. The first 48 attributes are of type `word_freq_WORD` and have real values in the range [0,100]. The next six attributes are of type `char_freq_CHAR` and have real values in the range [0,100]. Three attributes are of type `capital_run_length_average`,

² <https://archive.ics.uci.edu/ml/datasets/Spambase>

capital_run_length_longest and capital_run_length_total; they measure the average, maximum and total lengths of uninterrupted sequences of capital letters in the message. The last attribute denotes whether the e-mail was considered spam (1) or not (0). There was no further information provided about the message because of end user privacy issue. LingSpam dataset³ was collected by a linguistic mailing list in 2000. It contains 2893 emails out of which 2,412 are non-spam email and 481 are abstracted versions of spam. The dataset creators used 10-fold stratified cross-validation to increase the confidence in their experimental findings. The header information was removed and subject line was kept. All words in the dataset were converted to lowercase. Markup and stop words were eliminated. Some words were substituted by their roots. Duplicated emails were removed from this corpora.

SpamAssassin⁴ is a large collection of raw messages publicly available by SpamAssassin. The corpus includes complete messages including email header and body. This helps evaluate the impact of using header, body or subject alone on the classifier performance. For privacy reasons, some hostnames have been replaced with "spamassassin.taint.org" (which has a valid MX record). However, in most cases the messages appear exactly as they were received. The spam emails were collected from various sources such as mailing lists or emails reported to SpamAssassin team. This dataset was used during the development of practical spam filter solutions including SpamAssassin. This dataset is updated more often than other datasets and the last time it was updated in 2006.

³ <http://www.aueb.gr/users/ion/publications.html>

⁴ <http://spamassassin.org/publiccorpus>

Enron⁵ has a large collection of non-spam messages. The emails were collected during the legal legislation of Enron Corporation. The email dataset was bought by MIT. The dataset was cleaned up by a research group at the Stanford Research Institute (SRI). Some messages have been deleted and attachments were removed. Invalid email addresses were converted to the form of user@enron.com. The dataset was originally made public, and posted to the web in 2003 by the Federal Energy Regulatory Commission during its investigation. It contains data from more than 150 users, mostly senior management of Enron. The dataset directory structure is shown in Figure 7. Each user represents a subdirectory and each contains several sub-directories (representing the categories).

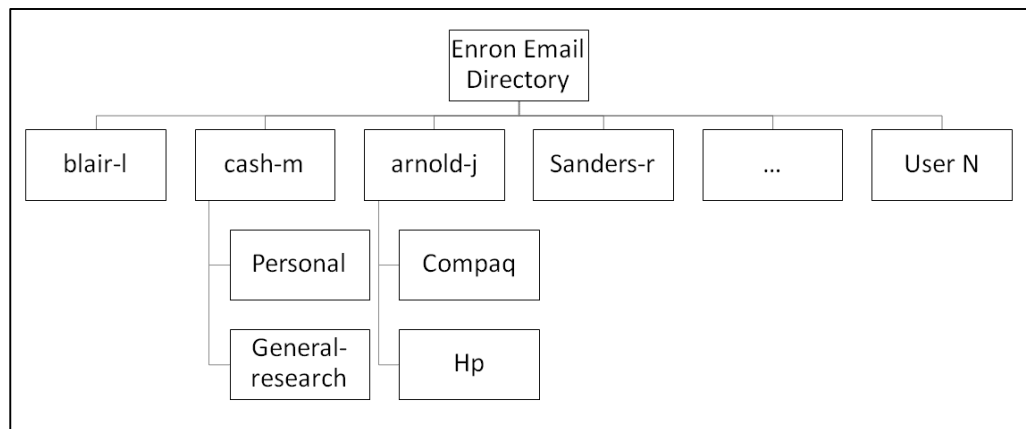


Figure 7: Enron email directory structure

PU123A⁶ Corpora contain personal and spam emails. They are available in preprocessed formats. Header fields and HTML tags were removed from the emails. However, subject

⁵ <http://www.cs.cmu.edu/~enron/>

⁶ <http://www.aueb.gr/users/ion/publications.html>

line has not been altered. The messages were converted to lowercase and strings of non-alphabetic characters were replaced with a single white space. To protect end-user privacy, each token was mapped to a unique integer. There are four datasets in this collection, namely PU1, PU2, PU3 and PUA.

Text REtrieval Conference (TREC) SPAM Track datasets (2005 – 2007)⁷ are the largest and most realistic datasets for evaluation. These corpora are publicly available on the web and can be downloaded by accepting the usage agreement. The first corpus was created for the TREC spam evaluation track based on Enron corpus and spam messages collected in 2005. The emails were classified and augmented by additional spam emails from public sources. The emails were split into four groups. The second release of the corpus was in 2006 and was in both English and Chinese. The English data version was collected from the web and publicly available spam whereas the Chinese version was collected from a mailing list and augmented with spam from a honeypot. The third release was in 2007 and combined one private and one public datasets.

ECML/PKDD⁸ 2006 dataset is a pre-processed dataset that are transformed into a bag-of-words representation . During preprocessing of this corpus, terms with occurrence less than 4 times were eliminated. For the preprocessing charset-, MIME-, base64-, URL- (RFC 1738), and subject line-decoding (RFC 2047) were used. For privacy reasons, the terms were encoded to protect end users.

ZH1⁹ is a Chinese dataset gathered by the Neural Language Processing Lab at Northeastern University. The messages were all simplified Chinese text encoded with GB2312/GBK.

⁷ <http://trec.nist.gov/>

⁸ <http://www.ecmlpkdd2006.org/challenge.html>

⁹ <http://homepages.inf.ed.ac.uk/lzhang10/spam/>

Since Chinese text is written continually without word delimitation, the text was segmented into words using a Chinese word segmenter developed at Northeastern University. After that, the messages were preprocessed to tokenize header fields, message body, sender name and recipient name.

GenSpam¹⁰ was gathered to overcome the problem of imbalance in number of spam and non-spam messages. The author collected the non-spam emails from 15 of his friends and colleagues. These emails are personal and commercial in nature. There are about 154k tokens in legitimate emails. The spam emails were sourced from sections 10-29 of the spamarchive collection, along with a batch gathered by the dataset author. The number of tokens in spam messages is about 281k tokens. The dataset was divided into training, validation and testing.

¹⁰ <http://www.benmedlock.co.uk/>

Table 2: Preprocessed spam email datasets

Dataset	Description	Dataset size (number of messages)			Related references	Creation Year	Headers included	Encrypted	Categories
		Spam	Non-spam	Total					
Spambase ¹¹	Constructed as bag-of-words that consists of 57 features	1,813 (39.4%)	2,788 (60.6%)	4,601	[6], [12]– [19], [82]	1999	No	Yes	Binary
PU1 ¹²	For privacy each distinct word replaced by arbitrary integer.	481 (44%)	618 (56%)	1,099	[20]	2000	No	Yes	Binary
PU2		142 (20%)	579 (80%)	721	[21]	2003	No	Yes	Binary
PU3		1,826 (44%)	2,313 (56%)	4,139	[21], [22]	2003	No	Yes	Binary
PUA		571 (50%)	571 (50%)	1,142	[21]	2003	No	Yes	Binary
ECML-PKDD 2006 ¹³	Messages represented as bag-of-words used for a competition	3,500 (50%)	3,500 (50%)	7,000	[23], [24]	006	No	Yes	Binary
GenSpam ¹⁴	Header information reformatted in XML	32,332 (78%)	9,072 (22%)	41,404	[9]	2005	No	No	Binary

¹¹ <https://archive.ics.uci.edu/ml/datasets/Spambase>

¹² <http://www.aueb.gr/users/ion/publications.html>

¹³ <http://www.ecmlpkdd2006.org/challenge.html>

¹⁴ <http://www.benmedlock.co.uk/>

Table 3: Raw spam email datasets

Dataset	Description	Dataset size (number of messages)			Related references	Creation Year	Headers included	Encrypted	Categories
		Spam	Non-spam	Total					
LingSpam ¹⁵	Collected from a linguistics mailing list	481 (17%)	2,412 (83%)	2,893	[25]	2000	No	No	Binary
SpamAssassin ¹⁶	Messages come from several sources	1,897 (31%)	4,150 (69%)	6,047	[20]	2002 (Updated 2006)	Yes	No	Binary
Enron ¹⁷	Collected from about 158 users, mostly senior management of Enron	13,496 (42%)	16,545 (58%)	30,041	[20] [26], [27]	2003	Yes	No	Multiple/ Binary
TREC 2005	Available online through NIST ¹⁸ and considered to be the largest dataset	52,790 (57%)	39,399 (43%)	92,189	[22], [28], [29]	2005	Yes	No	Binary
TREC 2006		24,912 (66%)	12,910 (34%)	37,822	[30]	2006	Yes	No	Binary
TREC 2007		50,199 (66%)	25,220 (34%)	75,419	[31]	2007	Yes	No	Binary
ZH1 ¹⁹	Chinese corpus	1,205 (74%)	428 (26%)	1,633	[32]	2004	Yes	Yes	Binary

¹⁵ <http://www.aueb.gr/users/ion/publications.html>

¹⁶ <http://spamassassin.org/publiccorpus>

¹⁷ <http://www.cs.cmu.edu/~enron/>

¹⁸ <http://trec.nist.gov/>

¹⁹ <http://homepages.inf.ed.ac.uk/lzhang10/spam/>

Two other datasets are related to short messages. The first one is “SMS spam corpus v.0.1 big” [10] (will be called SMS big in the rest of this thesis). This dataset is a collection of 1002 legitimate message and 322 spam messages in English language. The legitimate SMSs were randomly extracted from the National University of Singapore (NUS) SMS Corpus and the Jon Stevenson Corpus. The spam messages were collected from the Grumbletext website. The average length of words per message is 4.44 characters long and the average number of words is 15.72 . The other dataset is SMS UCI Spam Collection [11]. It is available in at UCI machine learning repository in a raw data format. This dataset is in English. It was collected in 2012 from four main sources: Grumbletext website (425 SMS), Caroline Tag's PhD Theses (450 SMS), National University of Singapore (3,375 SMS) and Jon Stevenson Corpus (1,324 SMS). It is considered the largest SMS spam corpus publicly available.

Table 4: Spam SMS datasets

Dataset	Dataset size (number of messages)			Creation date	Tokens per message	Related references
	Spam	Non-spam	Total			
SMS Big	322	1,002	1,324	2007	15.72	[10], [33], [34]
UCI SMS Spam	747	4,827	5,574	2011	14.56	[10]

2.1.2 Feature Extraction

Bag-of-words (BoW) scheme is the most widely used approach model for feature extraction because of its simplicity for classification purposes [38]-[40]. Each e-mail message is represented by a vector using vector space model (VSM). The vector could be represented as $X = (x_1, x_2, x_3, \dots, x_m)$, where m is the number of features (attributes). In the BoW approach, a vector could be created using different schemes as shown in Table 5. Each attribute will be given a weight to measure its importance to the document. A weight can be assigned to each attribute based on the occurrence of the term or existence of it. In the binary feature weight scheme, each element or attribute is either 1 or 0 depending on existence of the word or the feature. Term frequency (TF) scheme is also used to assign a weight to each attribute. In this approach the weight is assigned based on the number of occurrence of each term in the document. Sometime this weight is normalized by dividing the total number of terms in the documents. The normalization is performed because most of the time the processed documents are not of the same length. The third most widely used approach is TF-IDF (term frequency-inverse document frequency). In this scheme, each element represents the term frequency times a global parameter representing the inverse of the document frequency.

Table 5: Bag of word vector creation

Attributes	Description
Binary Term Occurrence (BTO)	$X_{ij} = 1$ if term exists 0 otherwise
Term Frequency (TF)	$X_{i,j} = \frac{TF_{i,j}}{\sum_K TF_{k,j}}$
Term Frequency–Invers Document Frequency (TF-IDF)	$X_{i,j} = TF_{i,j} \cdot \log \frac{ D }{ \{d_j : t_i \in d_j\} }$ where $ D $ is the number of documents in the corpus and d_j is a specific document.
Term Occurrence (TO)	$v_{i,j} = f_{i,j}$ The absolute number of occurrence of a term

Another approach combines internal and external information features [35]. In this approach, some features are extracted from the email itself such as the number of dots in URL links embedded in the message body. Other features are extracted from external information such as the age of “linked-to” domain names. In [36], structural features are used for detecting phishing emails. This approach includes features from style markers and structure attributes such as the structure of the subject line and greetings in the message body. Hybrid feature selection that combines behavior based and content based was proposed in [37]. An example of the features considered in this approach is analyzing the message-ID tag and sender email.

Table 6 is a summary of the main spam feature extraction methods. It also shows the datasets and algorithms that were used in addition to some performance results.

Table 6: Spam feature extraction

Type of Feature	Related references	Dataset	Preprocessing	Classifier	Experiment results	Remarks
BoW TF-IDF	[38]	TREC	-	SVM	AUC = 99.07%	Fuse information of words in an email by using TF-IDF formula
	[39]	SpamAssassin and a specially created repository	Stemming and stopping	KNN, SVM, NB	Accuracy above 90%	
	[40]	Two specially created repositories	-	RIPPER, SVM	Error rate reach to 1.8%	1000 best features and another data set where the dimensionality was over 7000
Semantic Ontology concept by (TFV) method	[41]	Enron-spam	Eliminate HTML tags, and stemming	NB	Spam F-measure is 94.87 %	Level of accuracy is low compared with other techniques; Proposed model works in 5 steps
Internal and external information	[35]	SpamAssassin	-	Random forest, SVM	False positive rate = 0.12%	10 features include WHOIS query
Structural features	[36]	A specially created repository	Remove all the blank lines	SVM	Accuracy reach to 100%	Features relevant to language, composition and writing
Online and offline features	[42]	TREC	-	SVM	Accuracy reach to 97%	16 features. Requires higher cost computation because of online features
Hybrid	[37]	SpamAssassin	-	Bayes Net algorithms	Accuracy reach to 96%	Content-based and behavior-based. Using 7 features

2.1.3 Feature Selection Methods

In this stage, the objective is to identify the features that are related to each class. Some features are correlated with the class label while others are irrelevant hence identifying the right features is important. Moreover, it is better to reduce the feature space dimensionality to avoid the over-fitting problem. Additionally, some classifiers cannot handle large number of features and their performance would be improved if only the right features are used. Figure 8 shows the original and reduced feature vectors. Reducing the feature space could be achieved using different methods such as stop-word removal, stemming, and feature selection techniques. In stop words removal, we determine common words (e.g. “the”, “and”) that are irrelevant to different classes. In stemming, the word is converted to its base form where suffixes and prefixes are trimmed. This is helpfully to avoid treating different forms of the same word and different attributes. For example, singular, plural and different tenses are converted to its stem form such as “moved” and “moving” are stemmed to “mov”. Porter stemming algorithm²⁰ is the most widely used stemming algorithm and it became the de facto standard algorithm for English language stemming [47]. For the feature selection techniques, there are three main categories: filter, wrapper, and embedded based methods.

²⁰ <http://tartarus.org/~martin/PorterStemmer/index.html>

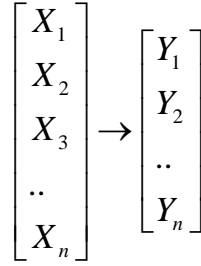


Figure 8: Feature Selection Process

2.1.4 Filter Feature Selection

In the filter feature selection approach, there are two basic components in the model feature search and feature selection criteria/evaluation as shown in Figure 9. Information gain (IG), mutual information (MI), gain ratio (GR), chi-square (χ^2), principal component analysis (PCA), and document frequency (DF) are examples of the widely used filter feature techniques [77]. The challenge in this model is to filter feature independently of the induction algorithm such as classification algorithm. The relevant features are selected in the preprocessing step. This model does not take into consideration the effect of the selected features on the classification performance. The advantages of this approach over the wrapper, which will be discussed in the next section, is that it does not require a lot of computing resources and it is less time consuming. During the preprocessing phase, features with the highest weight are passed to the classifier and used to build the classifier model. The number of the features (subset) that will be used to build the classifier model

is set by the user. Selecting the right number of features would affect the classifier performance.

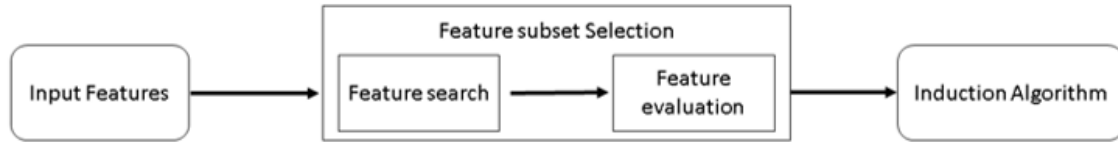


Figure 9: Filter feature selection model

Gini Index (GI) [52, 77] also known as (Gini ratio) is a popular approach for calculating the discrimination level of a feature. The value of GI is in the range $(1/k, 1)$, where k is the number of classes. Higher value of GI for a word indicates a greater discriminative power of that word. The computational time is $O(d w k)$, where d is the number of documents, w is number of words, and k is number of classes.

Information Gain (IG) [15, 76, 77] is widely used for measuring the goodness of various features in the field of machine learning. It is the expected reduction in entropy caused by partitioning the dataset instances according to a given attribute. The greater the value of the information gain for a specific word, the greater is the discriminatory power of the word. The computational time complexity is similar to GI.

Mutual Information (MI) [15, 77] measure is very common in statistical language modeling of word association. It provides a formal way to model the mutual information between the features and the classes. The time complexity of MI is similar to IG and GI computation.

The weakness in this approach is that the score is strongly influenced by the marginal probabilities of term.

Chi-square (χ^2) [15, 77] statistic measures independence between a word X and a specific class C . The value of χ^2 is zero if X and class C are independent. Both chi-square and MI are different methods of calculating the correlation between words and class. However, the advantage of chi-square over MI is that it is a normalized value which can be utilized to compare words in the same class. The time complexity of chi-square is similar to IG and MI computation.

Document Frequency (DF) [15] method ranks terms based on how they are rare or frequent across all documents. This method is the simplest technique among the feature selection methods. The higher the value of DF is, the more frequent the term is and the less information it provides.

Table 7 summaries the widely used methods in the literatures to calculate the feature weight.

Table 7: Features weights

Method	Formula
Gini Index (GI)	$G(w) = \sum_{i=1}^k P_i(w)^2$ Where w is a word
Chi-square (χ^2)	$\chi^2(X, C) = \frac{[P(x, c)P(\bar{x}, \bar{c}) - P(x, \bar{c})P(\bar{x}, c)]^2}{P(x)P(\bar{x})P(c)P(\bar{c})}$ Where X is a word and C is the class
Mutual Information (MI)	$MI(X = x, C = c) = \log \frac{P(X = x, C = c)}{P(X = x).P(C = c)}$ Where X is a word and C is the class
Information Gain (IG)	$IG(X, C) = \sum_{x,c} P(X = x, C = c). \log \frac{P(X = x, C = c)}{P(X = x).P(C = c)}$ Where X is a word and C is the class
Document Frequency (DF)	$DF(X) = \{d_j : X \in d_j\} $ Where d is a document and X is a word

2.1.5 Wrapper Feature Selection

The second approach that is used to select features is the wrapper model [78]. This approach depends on the classifier since it evaluates the features based on the classifier performance. As illustrated in Figure 10, there are three components in this model: feature search, feature evaluation and classification algorithm. This approach is very slow, requires high computational cost for space search, and has the potential to over-fit training data. However, it provides higher accuracy than the filter selection method. Examples of this approach are the backward, forward and genetic algorithms. For instance, Figure 11 shows the forward feature selection algorithm [78].

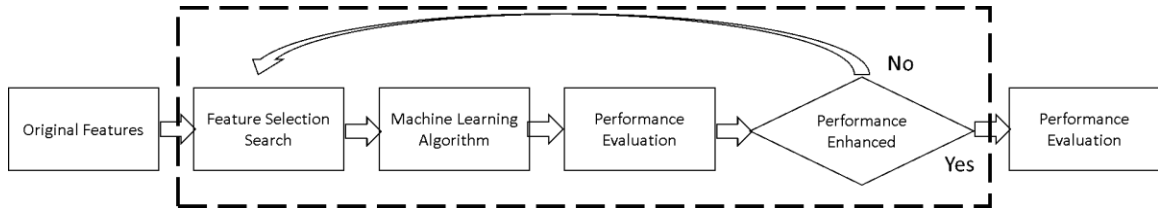


Figure 10: The wrapper approach for feature selection

1. *Initiate feature set* $Y_0 = \{\emptyset\}$; $k = 0$;
2. *Select the next best feature* $x^+ = \arg \max_{x \notin Y_k} [J(Y_k + x)]$;
3. *Update* $Y_{k+1} = Y_k + x^+$; $k = k + 1$;
4. *Go to 2*

Figure 11: Forward feature selection algorithm

2.1.6 Embedded Feature Selection

In Embedded Feature Selection, some classification algorithms, such as random forests and decision trees, have built-in feature selection. In this approach the classifier automatically selects and identifies the best features while developing its model. The classifier as part of the learning phase performs variable selection to identify the optimum features. The advantage of this model is that it is less computationally intensive compared to the wrapper approach. however, its performance depends on the used classifier [43], [44].

2.2 Machine Learning Algorithms

In the past, several research works on spam filtering have been conducted utilizing techniques such as Naïve Bayesian classifiers, decision trees, neural networks, etc. This section describes a number of selected research contributions related to email spam detection techniques.

2.2.1 Support Vector Machine (SVM)

SVM is considered one of the effective Machine Learning (ML) techniques used to solve classification problems [83]. It determines a decision boundary to divide the data space with the use of linear or non-linear separator between the different classes. Figure 12 shows how SVM separates two classes denoted by ‘x’ and ‘o’. The separator or decision surface, also called hyperplane, segregates the two classes. The points that are close to the hyperplane are called support vectors. SVM identifies the optimal boundaries (i.e. maximize the margin or distance) between the different classes to be used for classification. SVM is one of the commonly employed techniques to solve the spam problem. It classifies email by nonlinear mapping of the training dataset into a higher-dimensional feature space using kernel function or kernel mapping function. Then, it identifies the hyperplane that would maximize the distance between the two classes. One of the advantages of the SVM is its ability to learn independently of the feature space dimensionality.

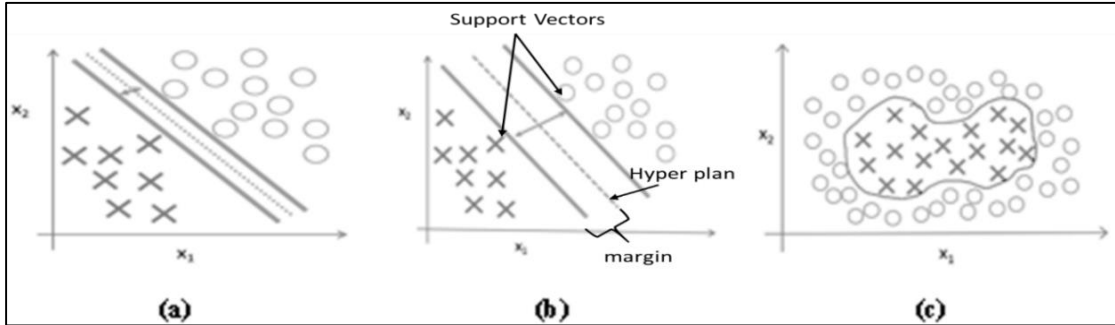


Figure 12: SVM binary classification (a) small margin, (b) large margin, (c) non-linear ²¹

2.2.2 Bayesian Learning

This approach builds a probabilistic classifier based on modeling the underlying message feature in different classes [15]. It is based on Bayes rule which is described mathematically as follow:

$$P(A = a | B = b) = \frac{P(A = a)P(B = b | A = a)}{P(A = a)P(B = b | A = a) + P(A = \neg a)P(B = b | A = \neg a)} \quad (2.1)$$

where $P(A = a)$ is the prior probability that a randomly picked message will be of type A , $P(B = b)$ is the probability that a randomly picked message will be represented by the feature vector $B = b$, and $P(B=b | A=a)$ is the conditional probability that a randomly picked message of type $A = a$ will have the representation of $B = b$ [15].

²¹ coursera.org (Machine Learning course by Stanford University)

2.2.3 Lazy Learning

Lazy-learning methods are instance-based techniques. The learning phase of this approach is delayed until a new email needs to be classified. An example of this technique is k -NN (nearest neighbor) classifier. Given an email, k -NN retrieves the k most similar instances from pre-defined learning examples. The similarity or distance of the neighbors is defined by some distance measures. The drawback of this approach is that it cannot detect spam when the reference dataset is huge.

2.2.4 Rule Based

A rule-based technique is another type of content classification. This technique is a knowledge-based technique that observes the presence of certain patterns and meta-data within an email. Several patterns that are usually associated with spam are specific words and phrases, uppercase and special character distribution, and malformed headers. The left-hand side of the rules belong to a word pattern whereas the right-hand side is the class label. Scores are assigned based on the pattern and if the score is high, then the email is an undesired email. This approach depends only on a set of rules and is usually augmented with other spam detection techniques [79]. The rule-based approach has the disadvantage of high false positive rate.

2.2.5 Others

Various proposals were made to improve or replace the current email protocol. The existing Simple Mail Transfer Protocol (SMTP) has a number of drawbacks, e.g. it does not check the identity of the message source. The researchers proposed to list the authorized outbound mail servers by domain owner so the email recipients can check if the email is coming from a legitimate source [80]. However, this approach could be bypassed and overcome by using spoofed IP addresses.

Establishing a small payment for sending a message was proposed to stop spam emails. In this approach [6], the senders have to pay a small fee to the email recipient for each email. In this case, the spammers would not send tremendous number of emails because they would be very costly. Common and normal email users would not be impacted as the number of sent emails usually equals the number of received ones .

Cryptographic authentication technique has been used to detect spam. In this approach, each email is augmented with digital signature of itself and its sender [72]. The email receiver queries the email sender for public key of the email to authenticate the email and its sender. Signature based filtering approach compares incoming email with the spam emails that have been seen before. The filter computes “signatures” for them, to identify the similarity. Since spam is sent in thousands if not millions, performing a similarity test can be used to detect spam. This approach will not block legitimate emails. However, it is not viable since a minor change in the email results in a different signature which makes this approach easy to bypass.

A blacklist, which can be DNS-based, email-address-based or IP-based, was also investigated [72]. This technique depends on collecting and storing the sources of spammers in a corpus to filter spam email based on this information. The legal e-mail server checks the corpus then denies accepting emails if the sender is blacklisted. This approach is very useful at the ISP level, but it has some limitations: (1) it is very difficult to keep the blacklist up-to-date, and (2) administrating the blacklist is costly.

A spam model based on Random Forests (RF) was proposed using parameters optimization and feature selection simultaneously [45]. This work optimizes two parameters: the number of variables in the random subset at each node (m_{try}) and the number of trees in the forest (n_{tree}). The objective is to maximize the classification accuracy.

Collaborative content filtering was evaluated and researched [74]. Here, many users share their judgment about what is desired mail and what is considered undesired mail. Every time the user receives an email, a special application suggests whether it is spam or not. This approach was proposed because of large number of spam characteristic. The major characteristic is that agents can interchange knowledge about spam.

A more advanced collaborative spam filter was experimented [73]. In this approach, a collaborative spam filtering system is utilized to enhance the individual spam detection ability. Each client runs a different algorithm. The final decision is made based on a voting mechanism (as an example). The used algorithms and methods are Fisher's probability combination method, neural networks, naïve Bayes and decision trees. The classifications assigned by each approach are then linearly combined, with the weights of the classifiers that agree or disagree with the overall classification result. Utilizing UCI spam dataset, the

method achieved better results than the single method in isolation, with reduced false positives. The authors argued that the proposed model has important advantages, such as robustness to failure and easy implementation in a network.

In [46], Symbiotic Filtering (SF) approach was proposed to enable the exchange of related features between different users to enhance local anti-spam filters. This work is based on a Content Based Filtering (CBF) and naïve Bayes learner. The main objective of this approach is to share information about what each local CBF has learned. Within SF there are two sharing possibilities either CBF models or relevant features. Both CBF and SF use Evolutionary Algorithms (EAs) for feature selection and Multinomial Naïve Bayes (NB) variants.

Ontology-based approach is one of the approaches to detect spam [46]. Three types of resource objects were used to store all kinds of information: information, data sources ID (user or website supplier) and information details (text or image or video). The developed module executes ontology reasoning and semantic similarity computation for the samples in the spam ontology samples database and newly acquired information. It determines the spam probability according to the calculation results. To block messages, the module filters them using three kinds of information: “sending number’s information” e.g. user brand and balance, “sending information of sending number” e.g. sending time segment, and credibility of sending number such as spam credibility of sending number .

Table 8 gives an over view of the widely used classifiers in the literature and which part of the email it analysis to classify the message. The table also shows the advantages and disadvantages of each method.

Table 8: Comparison of classifiers

Method	Email segment/part	Advantages	Disadvantages	Used in
Rule-Based Learning (RIPPER)	Body	<ul style="list-style-type: none"> Fast training and classification 	<ul style="list-style-type: none"> Easily bypassed High false positive rates 	[48], [49]
<i>k</i> -Nearest Neighbor	Body Header	<ul style="list-style-type: none"> No explicit training 	<ul style="list-style-type: none"> Sensitive to the value of <i>k</i> Can encounter problems in very high dimensional spaces 	[50]
Bayesian	Body Header	<ul style="list-style-type: none"> Quick training and classification 	<ul style="list-style-type: none"> Bayesian poisoning 	[50]
Neural Networks	Body Header	<ul style="list-style-type: none"> Fast to run Handles noisy data well 	<ul style="list-style-type: none"> Long training time 	[49]
Support Vector Machines	Body Header	<ul style="list-style-type: none"> Execution speed is very fast Easy to integrate with end user interactive feedback approach. Easy to implements 	<ul style="list-style-type: none"> Training time increases exponentially with the number of training elements involved Prone to overfitting and thus bad generalization Perform very badly if the features are highly correlated. 	[2], [20], [46]
Boosting (AdaBoost)	Body Header	<ul style="list-style-type: none"> Seems not to over-fit in practice Very simple to implement 	<ul style="list-style-type: none"> Unable to handle weak learners with an error rate greater than 1/2. Long training time 	[51]
SMTP-path Analysis	Header	<ul style="list-style-type: none"> Fast classification 	<ul style="list-style-type: none"> Easy to bypass by using a new IP source (spooof IP address) 	[53]
Social Networks	Header	<ul style="list-style-type: none"> Fast classification 	<ul style="list-style-type: none"> Misclassification of new sender 	[54]

2.3 Biological Background

A number of computational algorithms have been inspired by the biological immune system. These innovative algorithms are becoming integral part of the growing field in computer science known as artificial immune system (AIS) [55]. The artificial immune system is considered as a branch of computational intelligence as shown in Figure 13. The evolution of AIS area is illustrated in Figure 14.

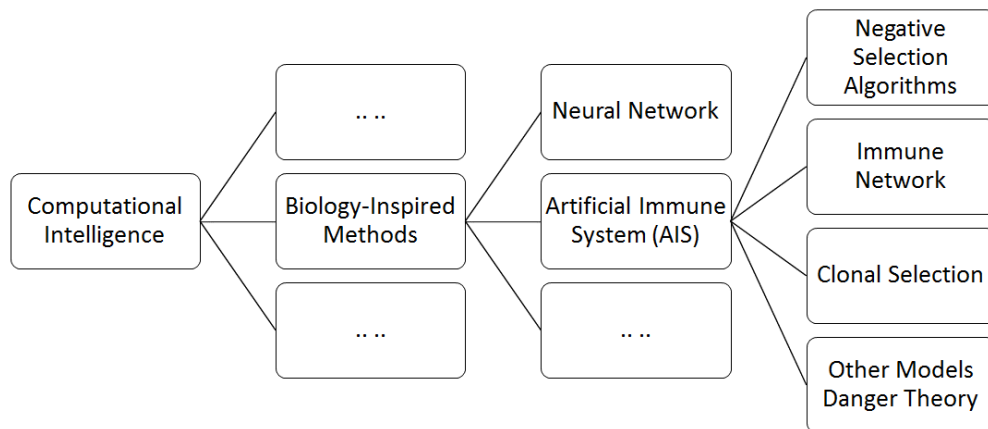


Figure 13: Artificial immune system as a branch of computational intelligence

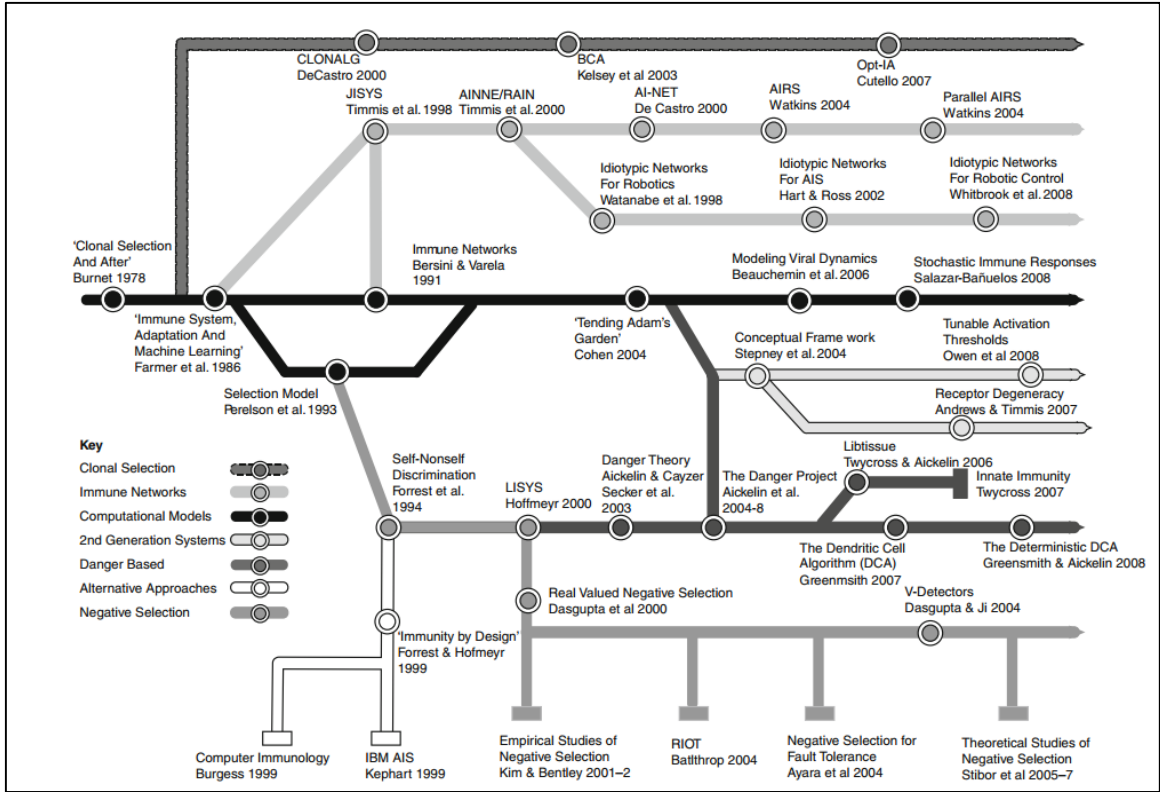


Figure 14: The evolution of AIS [56].

The major players of the human immune system are shown in Figure 15. This system consists of three protection layers as shown in Figure 16. The first layer is the physical barriers (e.g. skin in the human body). The other two layers are the innate and adaptive immune systems. Figure 17 depicts the type of cells that belongs to innate immune system and adaptive system.

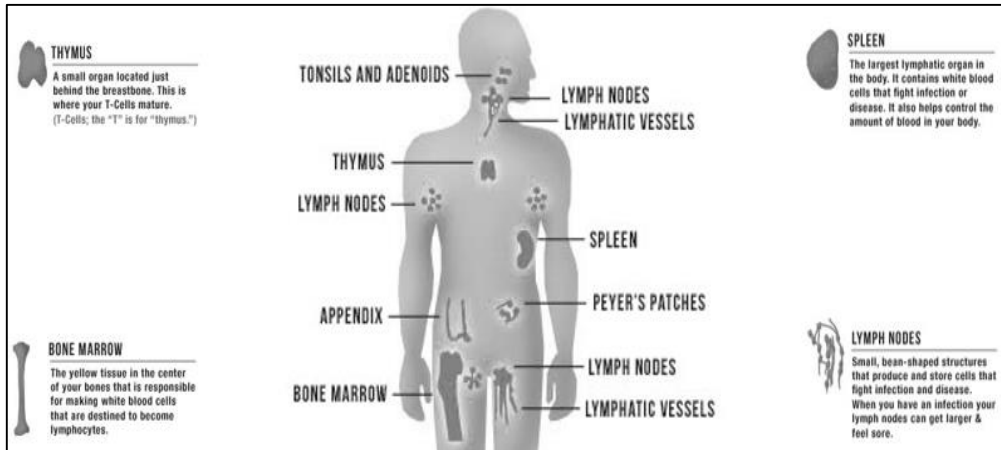


Figure 15: Human immune system locations²²

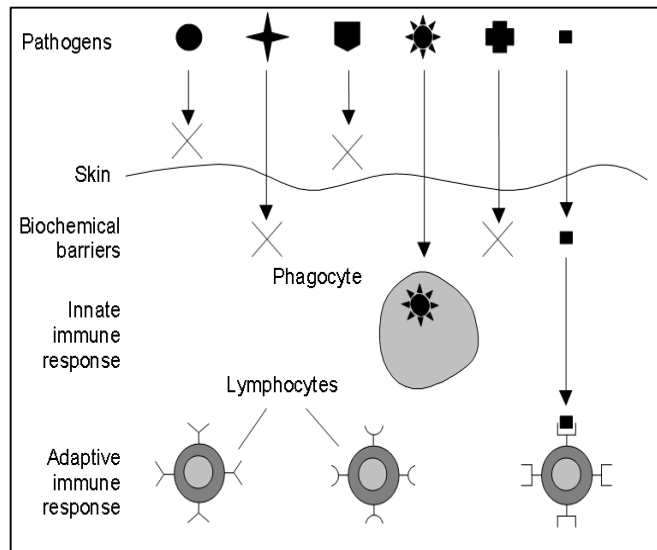


Figure 16: Human immune system layers

²² <https://www.aids.gov/hiv-aids-basics/just-diagnosed-with-hiv-aids/hiv-in-your-body/immune-system-101/>

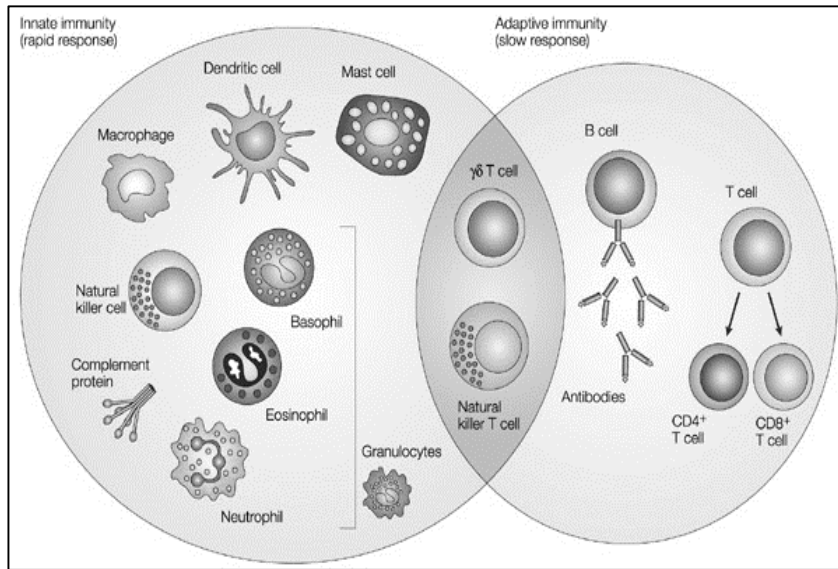


Figure 17: Innate and adaptive immune system cells²³

2.3.1 Innate Immune System

Innate Immune system reacts quickly to invaders (virus, bacteria, parasite, or to another person's tissues). It eliminates attacker in a period of time that can be minutes or hours. This is because the system does not differentiate between attackers. It only identifies non-self or self. There is no learning process in this system. The reaction does not change over the lifetime of the human. This system is very powerful. However, some microbes have learned to pass thought it. Therefore, there is a system called adaptive immune system which handles such type of attack.

²³ http://www.nature.com/nrc/journal/v4/n1/fig_tab/nrc1252_F1.html

2.3.2 Adaptive Immune System

The adaptive immune system is activated by the innate immune system to present response to specific pathogens. The adaptive immune system can learn and adapt its behavior to various types of pathogens [57]. This system is more complex and more advanced than the previous system. It evolves and adapts itself to handle new attackers.

The other capability that this system has is memory. This feature enables it to identify invader and recognize whether such invader has been identified previously. Hence, a formerly developed defense is reloaded and utilized to eliminate this invader quickly as shown in Figure 18. B-cells and T-cells are highly specialized defender cells of this system.

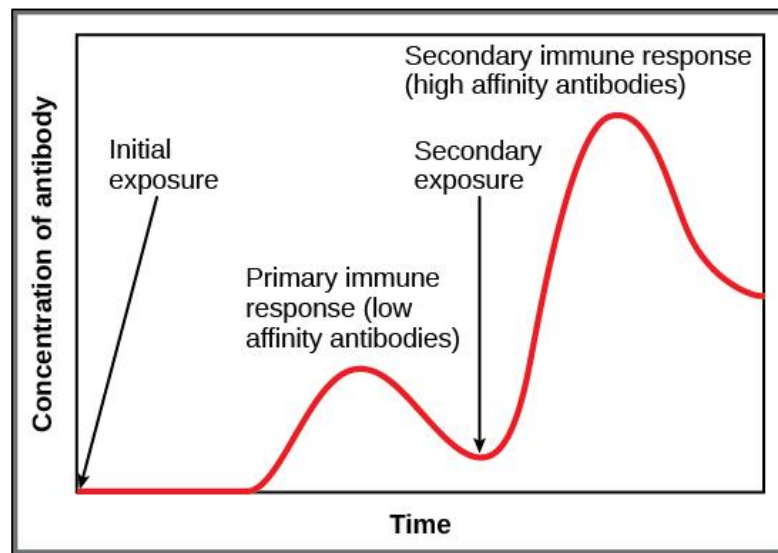


Figure 18: Adaptive immune system response time²⁴

²⁴ <https://www.boundless.com/biology/textbooks/boundless-biology-textbook/the-immune-system-42/adaptive-immune-response-234/immunological-memory-878-12128/>

2.4 AIS Algorithms and Applications

Artificial Immune System (AIS) was developed to mimic the human immune system. AIS categorization is based on the features of the immune systems that stimulate them. There are four main AIS approaches: Negative Selection Algorithm (NSA), Clonal Selection Algorithm (CSA), Algorithms based on idiotypic network model, and Dendritic Cell Algorithm (DCA).

AIS algorithms are performing similar function to the natural immune systems, i.e. defending and maintaining the system or host that they are running into. There are two generations of the artificial immune systems [58]. The first generation includes the following algorithms:

- Negative selection: designed for change detection, two-class classification, and intrusion detection problems.
- Clonal selection: mainly applied to function optimization and pattern recognition problems.
- Immune network: leveraged in unsupervised clustering problem domains.

This generation mimics the simplest models of the human immune system. Therefore, it has considerable limitations when applied to realistic problems. For example, the negative selection algorithm might have issue with scalability. It also generates high number of false alarms when applied to the intrusion detection problem.

The second generation is the Dendritic Cell Algorithm (DCA) which was developed as part of the “Danger Project” [58]. The objective of this project was to bring together latest research related to immunology and AIS to enhance the results of intrusion detection solution.

2.4.1 Negative Selection Algorithm (NSA)

The Negative Selection Algorithm (NSA) is inspired by the T-cell behavior which belongs to the adaptive immune system. The adaptive immune system has ability to tell the difference between structures of its own (self) and foreign ones (non-self). Figure 19 shows the self and non-self pattern. A theory for “Self” and “non-self” was proposed based on this observation. T-cells are exposed to agents that are part of human. Agent that is part of human will not be attacked by trained T-cells. The immune system ensures that infectious agents are attacked by properly identified T-cells.

This algorithm can classify agents as self or non-self by trying to match T-cells with each pattern that is being classified. If none of the T-cells categorizes the pattern as non-self, then that agent is identified as self. This algorithm needs a training phase. During the training phase, several patterns are exposed to the classifier. If a T-cell mistakenly identifies a pattern as non-self, it will be rejected. After completing the training, the self-pattern (originated from the body) identifies self and non-self (coming from outside). The algorithm works as follows:

- The system initializes a certain number of random detectors called “naïve detectors”.
- A self-set is produced by the system from the training data to contain data instances that are normal.

- The initialized naïve detectors in the first step are compared with the self-set to generate detectors that react to attacker.
- An alert will be triggered if the detector reacts to incoming instance.

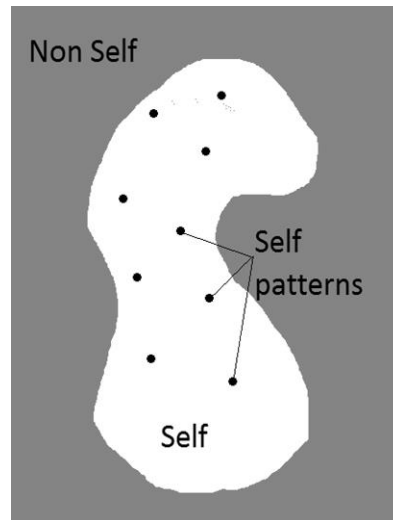


Figure 19: Non-self vs. self-pattern²⁵

2.4.2 Clonal Selection Algorithm (CSA)

The CSA was derived from Brunett’s theory of immune memory and clonal selection. It includes immune cells called B-cells [81]. This type of cells generates antibodies capable of recognizing diverse and numerous patterns of attacks or threats. Memory cells which are part of the immune memory can remember previously seen threats. This algorithm is applied to optimization and pattern recognition domains. The CSA algorithm works as follows [81]:

²⁵ http://cse-wiki.unl.edu/wiki/index.php/Artificial_Immune_Systems

- A number of random solutions to a given problem is initialized by the system.
- The initialized solutions are exposed to training.
- Based on a similarity measure, the best solutions are selected to generate several clones.
- A high frequency transformation process is conducted on the current solution.
- Memory cells are formed the best solutions.
- The previous steps are repeated until termination point is reached.

2.4.3 Idiotypic Network Based Algorithms

The third approach is using idiotypic network based algorithms. These algorithms were developed based on the theory of immune network by Jerne [81]. There are some similarities between this approach and CSA. It also shares properties with artificial neural networks. The immune system in this approach is seen as a network where immune elements communicate and work together even with the absence of antigens. The antibodies represent the nodes and the training algorithm involves growing or pruning edges between the nodes based on affinity. Immune network algorithms have been used in clustering, data visualization, control, and optimization domains.

2.4.4 AIS Applications

There are several applications for AIS algorithms due to their promising results [58, 59, 60, 62, 81]. A common application is to use the AIS algorithms to solve the Intrusion Detection problems to monitor network and detect network attacks by examining several symptoms. The number of login attempts, network traffic, and downloaded files sizes could be used as inputs to the solution. The system is considered under attack if the level of

danger is above a predefined threshold. Another application of AIS is detecting viruses in computers utilizing negative selection algorithm. The infected files can be recognized by T-cells. By leveraging this technique, the system can improve itself and detect mutated viruses.

2.5 Biological & Artificial Dendritic Cells

Due to the evolution in our understanding of the human immune system, a second generation of the artificial immune system was introduced. The new generation was developed in collaboration with immunologists. It includes several properties of the immune system such as robustness, error tolerance, and self-organization. The Dendritic Cell Algorithm (DCA) is the main algorithm that was developed recently as a second generation [59]. It was developed based on the danger theory, specifically the function and role of dendritic cells. Several researches were conducted and proved that DCA has the potential to solve security problems. For example, it was used in intrusion detection such as ping scan, botnet, and SYN scan with high accuracy. Before explaining the DCA algorithm, we start by giving a brief background about classical and danger theories that explain the human immune system.

2.5.1 Classical Immune System Theory

The classical theory that explained the immunology was based on self and non-self paradigm [60]. This theory stating that the immune system has the ability to distinguish between the body's own cells, recognized as "self," and foreign cells, or "non-self." The body's immune defenses normally coexist peacefully with cells that carry distinctive "self" marker molecules. But when immune defenders encounter foreign cells or organisms carrying markers that say "non-self", they quickly launch an attack as shown in Figure 20.

This theory failed in explaining the immune system reaction to several cases; for example:

- The immune system does not react to foreign bacteria such as the bacteria that are coming from food for example, are considered as non-self (foreign).
- Useful auto-reactive action against self-molecules are generated by stressed cells.
- Over the lifetime the human body changes and thus self changes as well. Hence, the question arises whether defenses against non-self learned early in life might be auto reactive later.
- The immune system does not attack successful transplant (which is non-self).

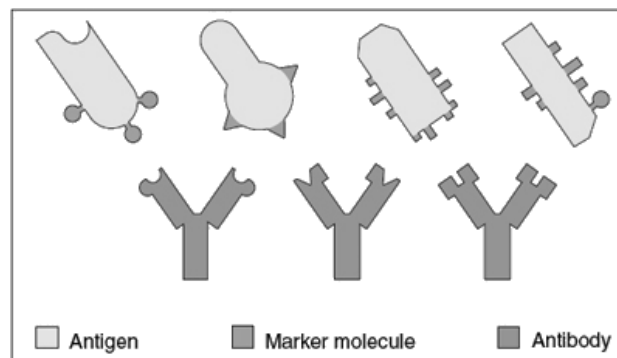


Figure 20: Antigens carry marker molecules that identify them as foreign²⁶

²⁶ <http://thyroid.about.com/library/immune/blimm02.htm>

2.5.2 Danger Theory

Due to the limitation of the old human immune system theory, researchers came up with another theory called danger theory [60]. This theory became popular over the past decade as it debates the old theory and points out that there is something beyond self and non-self distinction. It came up with the following conclusion about the immune system: it discriminates “some self from some non-self”. This theory describes the immune system as a system that does not attack non-self when it is harmless. However, it attacks self when it is harmful. This idea depends really on the idea of discrimination (which is the old viewpoint) except it reacts to danger not foreignness. In this theory, danger is measured by damage to cells indicated by distress signals that are sent out when cells die in unnatural death. Figure 21 depicts the danger theory model.

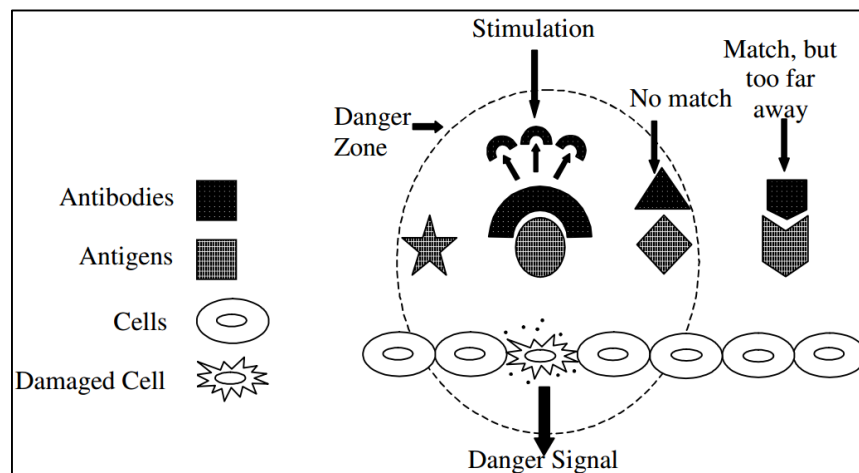


Figure 21: Danger theory model ²⁷

²⁷ <https://jackeylu.wordpress.com/2011/02/27/danger-theory-the-missing-link-between-artificial-immune-systems-and-intrusion/>

This theory was proposed by Matzinger and suggests that [60]:

- The immune system responds to signals of danger, rather than discriminating self from non-self.
- DCs gather debris in the tissue which are mixed with signals from the environment.
- Some of the suspicious debris produced by the attacker are called antigens.
- The danger signals are emitted by injured ordinary cells.
- DCs are responsible for the initial detection of invaders beside the induction of a variety of immune responses against such invaders.
 - The DC is able to analyze and integrate these signals to decide whether the environment is safe or dangerous.
 - If the signal is safe, the DC becomes semi-mature (as shown in Figure 22).
 - If the signal is dangerous, the DC becomes mature and instructs the adaptive system to respond.
- Antigen-presenting cells (APC) are activated via an alarm (danger signals).
- For T-cells to initiate an effective adaptive immune response, they require a co-stimulatory signal from APC.

There are three states of maturation of DCs as shown in Figure 23:

- Immature: collect parts of the signals and antigen (initial).
- Semi-mature: is an immature cell that internally decides that the local signals indicate safe.
- Mature: cell internally agree that the local signals indicate danger.

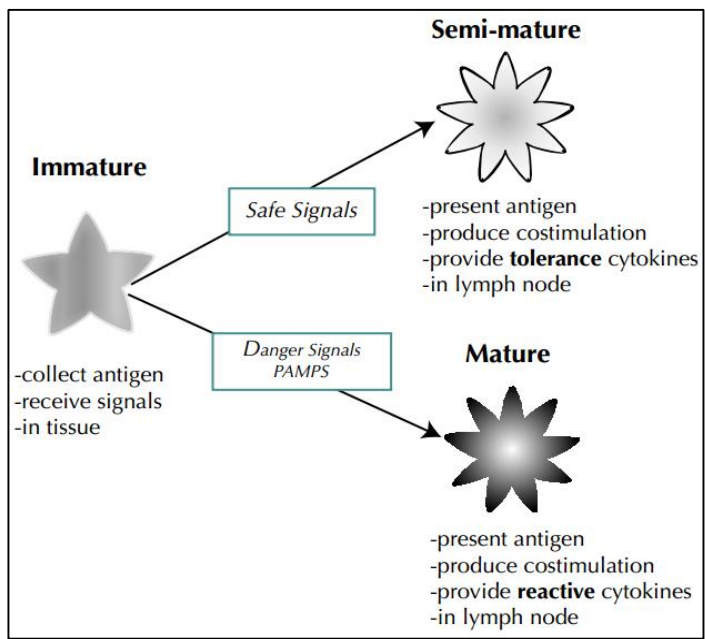


Figure 22: Transformation between DC states [60]

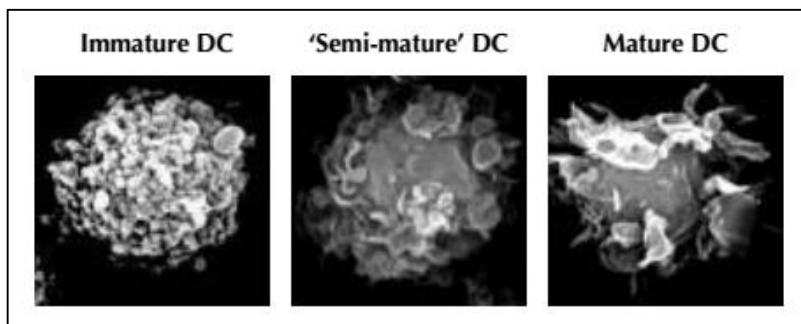


Figure 23: DC states [60]

Signals in DC are normally pre-categorized as:

- PAMP: an indicator to measure the existence of anomalous behavior, e.g. an event that harms the system. PAMP values increase as the observation of anomalous behavior increases.
- Danger: an indicator to measure the existence of a potential anomalous behavior. As the existence of a potential anomaly increases, the value of danger increases accordingly.
- Safe: an indicator to measure the confidence of a normal system behavior.

The DCs process two types of molecular information which are signals and antigens. From the local environment, the DCs gather the signals and indicators of the health of the monitored tissue (e.g. currently in distress or under attack). Knowing this information only could not help in identifying the source of the anomaly. Therefore, an antigen is required in order to link the evidence of the changing behavior of the tissue with the cause of this change in the behavior. In other words, the antigen is a potential originator of an anomaly and the classification of the antigen is based on the correlation between signals and antigens across the DC population.

2.5.3 Dendritic Cell Algorithm (DCA)

The development of the DCA algorithm follows the model shown in Figure 24. It was initiated as part of the danger Project [60]. The objective of the project was to bring together latest research from immunology and AIS to enhance the results of an intrusion detection system. The initial work presented by Greensmith was a DCA capable of finding danger signals within the standard Breast Cancer UCI Machine Learning data set [60].

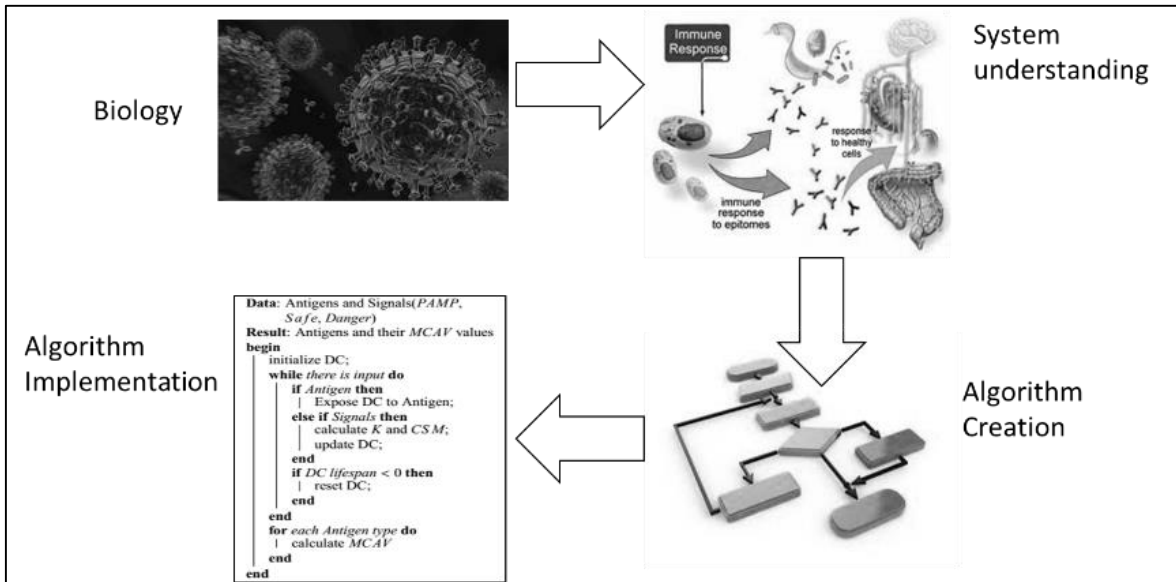


Figure 24: A depiction of the process used to develop the DCA

This algorithm correlates disparate data-streams in the form of antigens and signals then labels groups of identical antigens as “anomalous” or “normal”. There are two categories of input data:

- Signals: represented as a vector of numbers as measures of the monitored system’s status at a point of time.
- Antigens: represent the instances that need to be classified.

Several dendritic cells are deployed and work in parallel as detectors. Within the DC population, diversity is generated through the system lifespans. This would limit the quantity of information a DC object would process. During the DCs initialization phase, they are assigned different values for their lifespans. The DCA capability of detection is governed by the correlation of signals and antigens and the DC population diversity.

The dendritic cells receive three categories of signals (PAMP, danger and safe) and process them. Within the DCA algorithm, weights are manually predefined for each signal type. This controls the transformation of a cell based on input signals to output signals as shown in Figure 25. The status of the monitored systems is evaluated by the output signal. This is used to determine the presence of anomalies or misbehaviors.

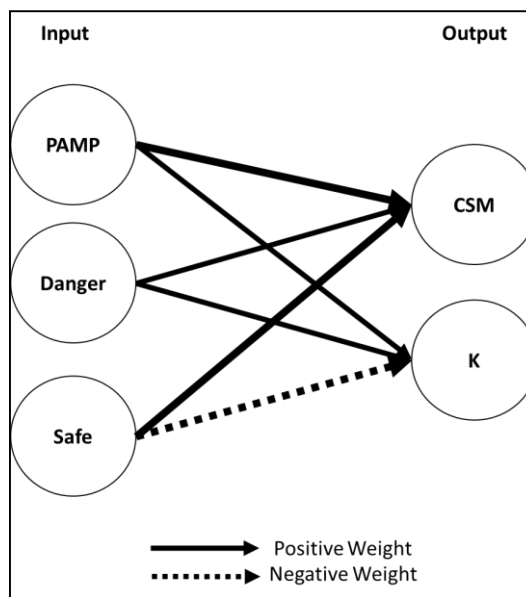


Figure 25: Signals weights and transformation to output

The final decision to classify an antigen is made based on the number of DCs that are fully mature. This could be done by voting or more advance approaches such as calculating a ‘mature context antigen value’ (MCAV) using the following equation:

$$MCAV = \frac{M}{Ag} \quad (2.5)$$

Where M is the number of mature cells within the antigen and Ag is the sum of the exposures to the antigen by those mature cells. $MCAV$ gives a probability of a pattern being an anomaly. The closer this values to 1, the greater the probability that the antigen is anomalous.

The algorithm processing is on three stages as shown in Figure 26: initialization, update, and aggregation. The initialization includes setting several parameters. In the next phase, the values of signals and antigens are updated each time new data appears in the system. Finally, the aggregation phase is initiated, where all collected antigens are subsequently analyzed and the $MCAV$ per antigen is derived. Figure 27 shows the algorithm pseudo code and Figure 28 illustrates the basics of its operation [60, 61].

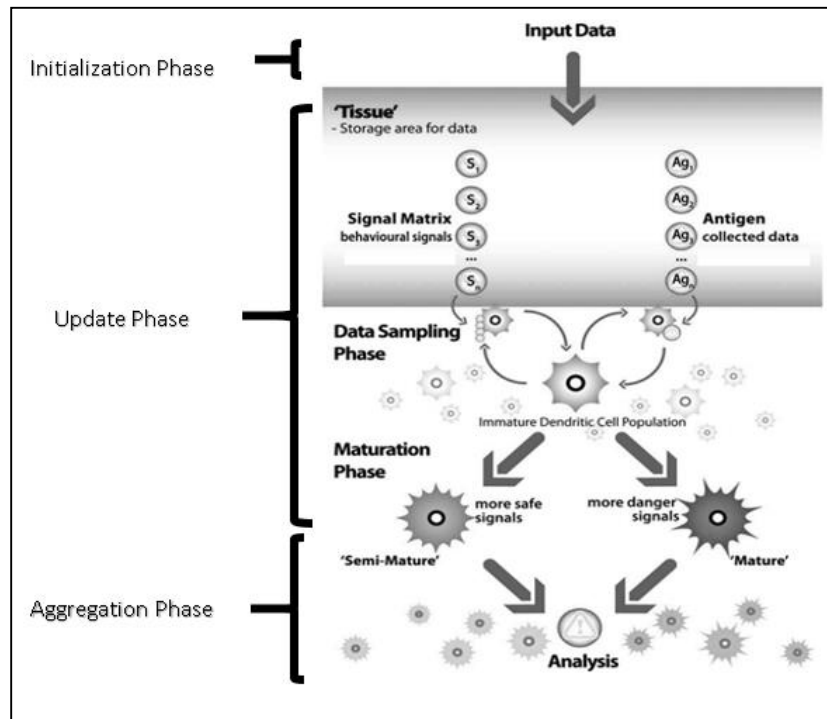


Figure 26: DCA three stages [61]

```

Data: Antigens and Signals(PAMP, Safe, Danger)
Result: Antigens and their MCAV values
begin
  initialize DC;
  while there is input do
    if Antigen then
      | Expose DC to Antigen;
    else if Signals then
      | calculate K and CSM;
      | update DC;
    end
    if DC lifespan < 0 then
      | reset DC;
    end
  end
  for each Antigen type do
    | calculate MCAV
  end
end

```

Figure 27: DCA pseudo code

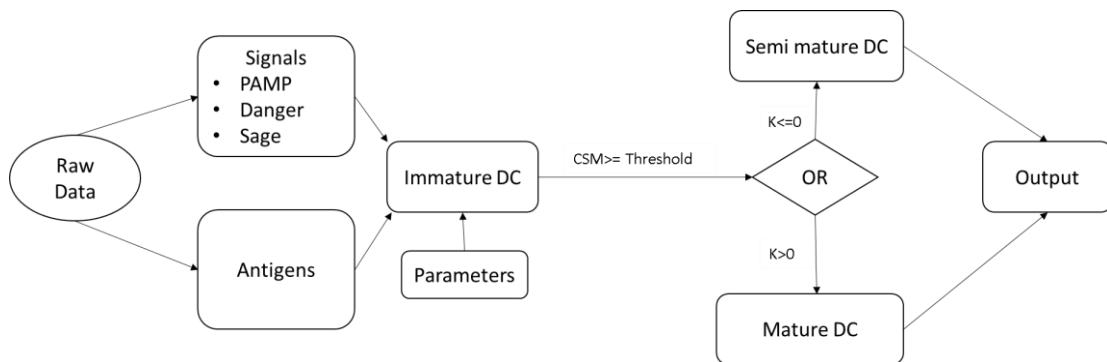


Figure 28: Decision of one cell

2.5.4 DCA Related Work

DCA has been implemented in several domains including plant classification, fraud detection, robotics security, botnet detection, port scanning detection, and more. The results show that the DCA is a promising solution. This section highlights examples of recent works that leverage the DCA.

In [62], DCA is used to classify plant leaves. The wavelet transform was used as a space feature and DCA as a classifier. Antigenes were represented by leaves images. Moreover, two types of signals were used: safe and danger. The safe signal was the distance between an unknown leaf texture features and known texture features called j . For the danger signal, it was represented by the distance between unknown leaf texture and \hat{j} unknown texture feature. The recognition result's accuracy was about 94%.

DCA was used to detect the break-in fraud for an online video on demand system [63]. Three groups of signals were used: PAMP, danger and safe. PAMP signals were represented by failures to order movies and danger signals were represent by successful movie order [63]. For the safe signals, three signals were used. The first two are associated with the login process and the third one is related to the billing notification. The results demonstrated that using the DCA to detect fraud has highly accurate decisions.

One of the successful applications of the DCA is botnet detection [75]. The DCA was used to detect the existence of a single bot on a compromised machine. Three types of signals were used to detect a bot. The PAMP signal was identified by the rate of change on invocation of selected API related to key logging activity whereas the danger signal was derived from the time difference between sending and receiving data. The time difference between outgoing consecutive communication function such as (send, send) was used for the safe signal. The experiments showed low rates of false positive and high rates of true positive.

CHAPTER 3

METHODOLOGY

In this chapter, we provide an overview of the proposed model of how email classification problem can be solved based on ideas from the Dendritic Cell Algorithm (DCA). We start with an overview of the system architecture. After that, we discuss the feature extraction of message content. Finally, we discuss the DCA classification technique.

3.1 Proposed Model Overview

In our proposed filtering model, we have considered two main parts of the email message: header fields and content. For SMS we considered the message content, but we enrich the content by adding tags. The system preprocesses the message content by removing HTML tags, stop words, and white spaces. After that, the system extracts different feature sets from the message content. Some features are extracted and selected based on opinions from a security expert, i.e. spam clues that an expert would normally look for. The system then uses a signal generator component to generate the required signals for the DCA algorithm. At the DCA fusion stage, the system classifies the input message based on its MCAV value. Figure 29 shows the system architecture of the proposed DCA classification system. The system consists five components: feature extraction, feature selection, signal generator, fusion and DCA algorithm. In the feature extraction there are two sub component one for extracting feature to filter and email and the other component is to extract feature for email classification. Figure 30 shows feature extraction process for spam filtering and Figure 31 illustrates feature extraction for email classification. The next step is to select most relative

features. Then, generate required signals and fuse them to right category. The last component is the DCA algorithm to classify an email to filter spam email.

In the next chapter, we will investigate the performance of our model on several datasets.

The next sections explain the system phases based on the proposed system architecture.

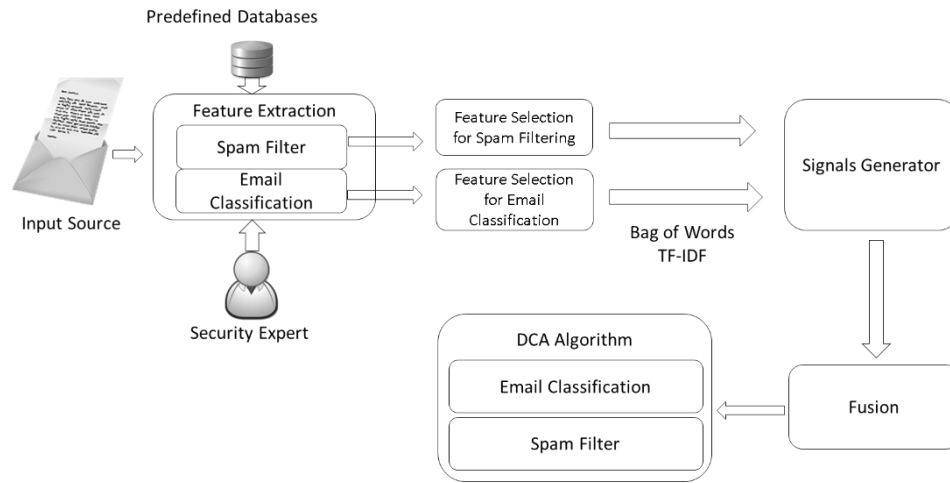


Figure 29: Layout of the proposed system

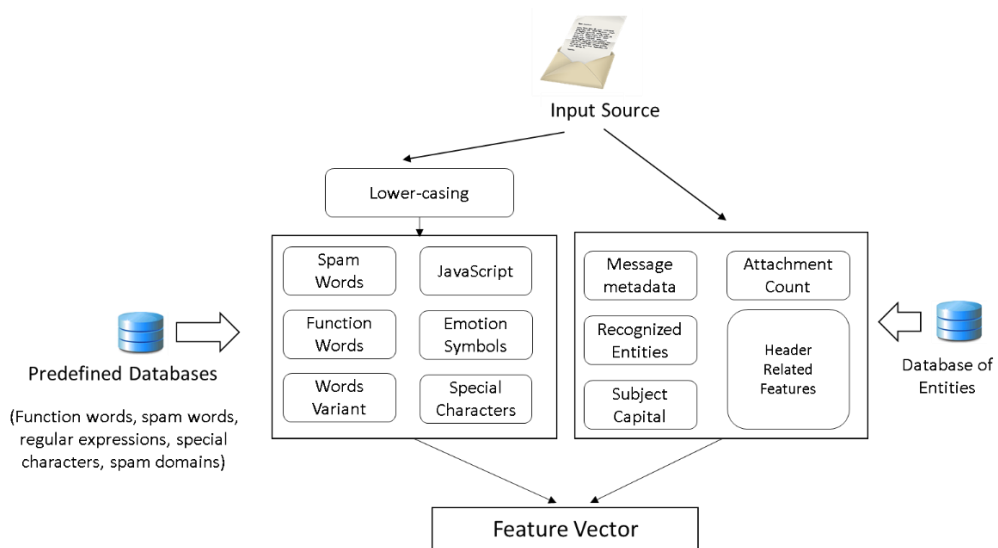


Figure 30: Feature extraction for spam filtering

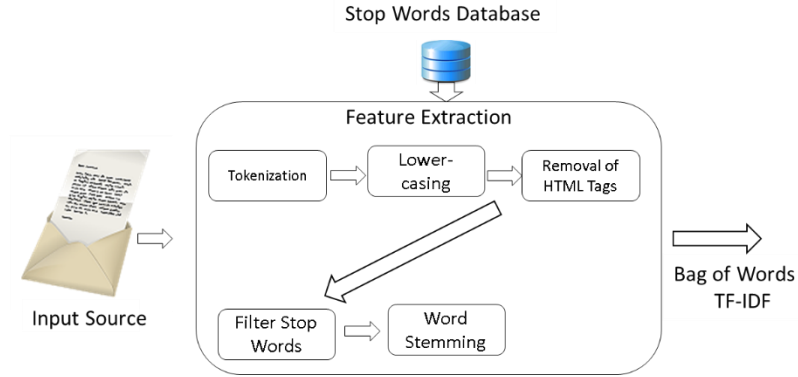


Figure 31: Feature extraction for email classification

3.2 Feature Extraction

There are two components within the feature extraction process. The first component is for spam filtering and the second one is for email classification. This process must be fast and should not require complex analysis; otherwise, it will delay the classification process. The features must be easily extracted from the message and empirically enhance the classification process.

We extract few features; however, their impact on the classifier's performance is significant. Novel features with low computing resources are extracted. These features mimic security expert practices to distinguish spam from legitimate. The extracted features are presented as a vector using Vector Space Model (VSM). The vector could be represented as:

$$Y = (y_1, y_2, \dots, y_m) \quad (3.1)$$

where m is the number of features (attributes). Each attribute will be given a weight to measure its importance to the message.

This step is considered the first step in the proposed model. Other processes depend on this step.

3.2.1 Spam Email Feature Extraction:

For the spam filtering process, some features will use lowercased tokens such as spam words and function word. The following features will be considered:

- Names of persons, organizations, money, and locations are recognized and tagged in the model as shown in the example in Figure 32. These tags are called “All_Tags”. We use OpenNLP to perform this process.
- Emotion symbols: We consider this feature since non-spammers use it very often in SMS message. Examples of these symbols are happy, angry or sad faces.
- Special characters: Spammers use special characters most of the time to encourage end-users to click on a link. For example, the dollar sign “\$” is used for winning a prize.
- Gappy words: They are words that have gaps between its characters such as “@” and “.”.
- Special words with different variations (e.g. misspelled) are also extracted. Examples of these words are prescription, price, and Viagra.
- JavaScript Code: Existence of Java Script code within the email body is a sign of phishing or spam. Hence, JavaScript code in the message is also extracted.
- IP address: We consider IP address as a feature since spam most of the time asks the user to click on a link to visit a website for a prize or to download an application. Moreover, the link directs the user to an IP address not to a domain name.

- Likely spam words: This is a set of terms that are most commonly used by spammers and are related to finance, dating, prizes, etc. [64]. We used the existence of terms listed in [64], [65]. We evaluated three lists of spam words:
 - The first list contains 17,000 spam words that we have collected from different sources. It is independent of the datasets on which our model is tested. It will be called Ind_17K_Spam_Words. This list consists of English and non-English keywords commonly used by spammers.
 - The second list contains 350 spam words that we have collected from different sources. It is also independent of the dataset on which our model is tested. It will be called Ind_350_Spam_Words. This list consists of English words that are widely used by email spammers.
 - The third list contains 250 spam words extracted from the dataset that we tested our model on. This list will be changed based on the dataset that we will test our model on. This list will be called Dep_250_SpamWords.
- Message metadata: This feature includes document length, which is the overall byte length of number of tokens and average token length.
- Function words (or grammatical words): These are words which have little lexical meaning or have ambiguous meaning, but exist to explain structural or grammatical relationships with other words within a sentence or specify the mood or attitude of the author. This feature is a closed class of words. There are relatively small and fixed number of function words, e.g. English language has around 300 function words. Function words are lexically unproductive and are generally invariable in form. Prepositions, pronouns, determiners, conjunctions, auxiliary verbs, and

particles are all considered function words. We evaluated function words feature because they are very unlikely to be subject to conscious control by author. This is due to their high frequency of use and highly grammatical role [66]. We leverage the word list available in [67]. Examples of function words are listed in Table 9.

- Blank-To: Spammers usually address the email to a group of users and put the names into the “BCC” (i.e., Blind Carbon Copy) field in the email.
- Number of characters in the “To” field. We used this feature as the spammer send the email to a randomly generated list of users.
- Number of users in the “To” field. We used this feature as spammer sometimes addresses the email to too several users.
- Number of users in the CC field. We used this feature as spammer sometimes addresses the email to too several users.
- Number of spam words in the subject fields. Spammer includes spam words in the subject fields to motivate the user to open the email quickly and click on the link.
- Spam domain: Several Internet domains have issues with their mail servers such as allowing sending email without authentication. We created a list of 200+ distrusted domains that are known of sending spam email.
- Subject capital: Using uppercase characters in the email subject would grab the end-user attention quickly; therefore, spammers use it.
- Subject length. This feature was explored because the spammer might use one word for the subject to grab the end user attention.

Table 9: Examples of function words

Class	Size	Examples
Prepositions	124	of, at, in, without, between
Pronouns	70	he, him, she, her, they, anybody, it, one
Determiners	28	the, a, that, my, more, much, either, neither, all, each
Conjunctions	44	and, that, when, while, although, or
Auxiliary and modal verbs	17	verbs be (is, am, are), have, got, do, shall
Particles	>86	no, not, nor, as, if, then, well, however, thus

pdate_Now - Double mins and 1000 txts on Orange tariffs. Latest Motorola/**Organization** , SonyEricsson/**Organization** & Nokia/**Organization** & Bluetooth FREE! Call MobileUpd8 on 08000839402 or call2optout/!YHL

Figure 32: Example of tagging for organizations

3.2.2 Email Multi-Category Classification Feature Extraction:

For the email classification, the email content is parsed, analyzed, and normalized to be presented into clear word format. At this phase, the message is prepared for the next stage by performing the following processes:

- Tokenization: Each email message is treated as a string and then divided into a sequence of tokens.
- Lowercasing of content: The entire message content is converted into lower case, so that capitalization is ignored.

- Removal of HTML tags: Many emails often come with HTML formatting. We remove all HTML tags so that only the text content remains.
- Removal of stop words: These are words that usually do not affect the context. They are removed from the message. Table 10 shows some examples of stop words that are removed by the model. For a complete list of stop words, refer to Appendix I.
- Word stemming: Words are reduced to their stemmed bases or root forms, e.g. removing all suffixes and prefixes such as “tion”, “ing”, and “er”. This process reduces the size of the feature set. There are several stemming approaches and algorithms such as table lookup approach, successor variety, n-gram stemmers, Porter stemming algorithm, and Paice/Husk stemming algorithm. In our work, Porter stemming algorithm is used since it is widely used [68].
- Removal of non-words: All white spaces (tabs, newlines, and spaces) are trimmed to a single space character.

Table 10: Sample of stop words

a's	able	about	above	according
accordingly	across	actually	after	afterwards
again	against	ain't	all	allow
allows	almost	alone	along	already
also	although	always	am	among

Table 11: Examples of rooted words using Porter Algorithm ²⁸

²⁸ <http://tartarus.org/~martin/PorterStemmer/index.html>

Word	Root
minding	mind
frames	frame
demise	demis
resolves	resolv
waverer	waver
lodge	lodg

3.3 Feature Selection

We used Gini Index (GI) approach for feature selection. It is also known as (Gini ratio). It is one of the widely utilized methods to calculate the discrimination level of a feature [69]. GI calculates the weight of each feature with respect to classification by computing the Gini index of the class distribution. The value of the GI is between $(1/k, 1)$, where k is the number of classes. The higher value of GI for a feature indicates a greater discriminative power of that feature. The computational time is $O(d \cdot w \cdot k)$, where d is the number of documents, w is number of words, and k is number of classes. The Gini index is calculated using the following equation:

$$G(w) = \sum_{i=1}^k p_i(w)^2 \quad (3.2)$$

where $p_i(w)$ is the conditional probability that a message belongs to class i given the fact that it contains the word w .

3.4 Signals Generator

This phase is an essential component in our model since the DCA depends on it to classify the input message. We developed two versions of this component. The first version uses

three machine-learning algorithms to generate the required signals. The second version uses two machine-learning algorithms and provides more accurate results with fewer resources and less computing time. This version is more efficient and scalable compared to the first version. In the upcoming sections, we will refer to the first version as 3c-DCA and to the second version as 2c-DCA.

1) 3c-DCA Signals Generator

In the initial proposed model, the three types of signals needed in DCA are generated using three machine-learning algorithms. The PAMP signal, which usually indicates an anomalous situation, is generated using the confidence level of a k -NN (k nearest neighbor) algorithm. The second type of signals is the existence of danger signal which may or may not present an anomalous situation. However, the probability of an anomaly is higher than under normal circumstances. In our experimental work, we employed the Naïve Bayes algorithm to generate the danger signal. Finally, the existence of safe signals indicate no anomalies exist. In our model, we utilized a Support Vector Machine (SVM) to produce the safe signal. The three signals are normalized between 0 and 1 before passing them to the DCA phase.

2) 2c-DCA Signals Generator

For this model, we used two machine-learning algorithms: Naïve Bayes (NB) and Support Vector Machine (SVM) algorithm. The PAMP signal indicates a high level of assurance of anomalous situation. It is generated using confidence level of one of the two classifiers if both agree with each other that the antigen is spam. The second type of signals is the existence of danger signals, which may or may not present an anomalous situation. However, the probability of an anomaly is higher than under normal circumstances. In our

experimental work, we used the average confidence level of the two classifiers if they disagree on the antigen classification. Finally, the existence of a safe signal indicates that no anomalies are exist. In our model, if the two classifiers agrees that the antigen is non-spam we utilize the confidence level of one of the two classifiers to be the safe signal. The pseudo code of this process is outlined in Figure 33. The two signals are normalized between 0 and 1 before passing them to the DCA phase.

```

Data:  $SVM_c$  and  $NB_c$  decisions;  $SVM_{cf}$ 
and  $NB_{cf}$  confidences
Result: signals:  $PAMP$ ,  $Safe$ ,  $Danger$ 
begin
|  $PAMP=0$ ,  $Safe=0$ ,  $Danger=0$ ;
| if  $SVM_c==NB_c$  then
| | if  $SVM_c=="Spam"$  then
| | |  $PAMP= \text{Max}(SVM_{cf},NB_{cf})$ ;
| | else
| | |  $Safe= \text{Max}(SVM_{cf},NB_{cf})$ ;
| | end
| else
| |  $Danger= \text{Avg}(SVM_{cf},NB_{cf})$ ;
| end
end

```

Figure 33: Process for calculating PAMP, safe and danger signals

3.5 Fusion Using DCA

Email messages (antigens) and signals for each email message are passed to the DCA as input. Moreover, each antigen is sampled multiple times using antigen multiplier to overcome the problem of antigen deficiency and to ensure it appears in different contexts [63]. The derived signals and associated antigens are presented to the DCA algorithm. The

DCA transforms the input signals into two output signals: *CSM* and *K* signals. If the *CSM* value is greater than a migration threshold of dendritic cells (which is randomly assigned), the dendritic cell stops sampling antigens and signals. *K* is the overall amount of abnormality of signals seen by a dendritic cell. These two output signals are calculated as follows:

$$\Delta CSM = P_n \times P_{csm,w} + D_n \times D_{csm,w} + S_n \times S_{csm,w} \quad (3.3)$$

$$\Delta K = P_n \times P_{k,w} + D_n \times D_{k,w} + S_n \times S_{k,w} \quad (3.4)$$

where P_n is the PAMP input signal value, D_n is the danger signal value, S_n is the safe value; $P_{csm,w}$, $D_{csm,w}$ and $S_{csm,w}$ are the weights related to the calculation of *CSM*; $P_{k,w}$, $D_{k,w}$, and $S_{k,w}$ are the weights related to the calculation of *K*. The mature context antigen value (*MCAV*) for each antigen type is calculated and compared with a threshold for final prediction. The *MCAV* value is calculated for each antigen type using the following formula:

$$MCAV = \frac{M_i}{\sum Ag} \quad (3.5)$$

where i refers to the antigen type (spam), M_i refers to the number of times that antigen appears in the mature context, and $\sum Ag$ is the total number of antigens. The *MCAV* value is used to classify an email by comparing it with a preset threshold calculated from:

$$at = \frac{an}{tn} \quad (3.6)$$

where at is the derived threshold, an is the number of anomalous data items and tn is the total number of data items. The classification rule is as follows:

$$i = \begin{cases} spam & \text{if } MCSV > at \\ ham & \text{otherwise} \end{cases} \quad (3.7)$$

Figure 34 illustrates the DCA algorithm pseudo code.

```

Data: Antigens and Signals(PAMP, Safe, Danger)
Result: Antigens and their MCAV values
begin
  initialize DC;
  while there is input do
    if Antigen then
      | Expose DC to Antigen;
    else if Signals then
      | calculate K and CSM;
      | update DC;
    end
    if DC lifespan < 0 then
      | reset DC;
    end
  end
  for each Antigen type do
    | calculate MCAV
  end
end

```

Figure 34: Pseudo code for DCA

CHAPTER 4

EXPERIMENTS

This chapter presents the results and discussions of our research findings. We first describe the adopted datasets in our work and the measures to evaluate the proposed model. The experiments and discussions have been divided into three parts. The first part discusses the experiment of using the first version of DCA (called 3c-DCA). The second part discusses the second version of DCA (called 2c-DCA). Finally, the third part discusses the general email classification problem using 2c-DCA.

4.1 Experiments Setup

We performed several experiments to compare our proposed model with well-known machine-learning algorithms such as Naïve Bayes (NB), SVM, and k -NN. The experiments were conducted on several datasets that are listed in the next section. RapidMiner version 6.5 was utilized to measure the performance of the machine-learning algorithms. Default configurations of the algorithms were used in our experiments as shown in Table 12. We used a machine with 8 GB RAM, 2.4 GHz CPU, and Microsoft Windows 8.1.

Table 12 Machine Learning Configuration

Algorithms	Parameters
K-NN	$k = 1$
Support Vector Machine	Max Iteration = 100000

4.2 Datasets

To evaluate our work we used five datasets for spam filtering and one dataset for email classification. Table 13 provides a summary of the used datasets and their usage in our work. These datasets are explained in Section 2.1.1 (Table 2, Table 3 and Table 4).

Table 13: Used Datasets

Dataset	Dataset size (number of messages)			Used in	Dataset Type
	Total	Spam	Non-spam		
Spambase	4601	1813	2788	Spam Filtering	Email
SpamAssassin	9324	2387	6937		
TREC2005	92,189	52,790	39,399		
UCI SMS Spam	5,574	747	4827		SMS
SMS Big	1,324	322	1,002		
Enron	30,041	13,496	16,545	Email Classification	Email

4.3 Evaluation Measures

Several performance measures were used which could be calculated online, using real measurements, or offline, using pre-collected and classified email messages. Offline provides quick insights about the filter effectiveness.

The classifier results were compared with other model results using True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) measures. In this context, a positive means spam. For example, a “true positive” is a spam message which is correctly classified as a spam and a “false positive” is a non-spam message which is wrongly classified as a spam. Confusion matrix or contingency table is used to visualize the classifier performance as shown in Table 14. This matrix focus on the capability of the classifier to predict an input without taking into consideration the model performance. The matrix

Table 14: Confusion matrix

		Predicted	
		Spam (Positive)	Legitimate (Negative)
Actual	Spam (Positive)	TP	FN
	Legitimate (Negative)	FP	TN

Moreover, over all filtering accuracy metric (Acc), Precision, Recall, and AUC were used to measure the performance of the classifier. Below is a definition of these measures [70]:

- Accuracy: measures the effectiveness of the classification model based on the fraction of correctly classified email messages and is given by:

$$Acc = \frac{TP + TN}{TP + FP + FN + TN} \quad (4.1)$$

- False Negative Rate (FNR): measures the proportion of spam email messages that are falsely classified as legitimate and is calculated using the following formula:

$$FNR = \frac{FN}{Total\ Positive} = \frac{FN}{TP + FN} \quad (4.2)$$

- False Positive Rate (FPR): measures the proportion of legitimate e-mails that are falsely classified as spam and is calculated using the following formula:

$$FPR = \frac{FP}{Total\ Negative} = \frac{FP}{TN + FP} \quad (4.3)$$

- Spam Precision (SP): measures the level to which rejected messages by the spam filter are truly spam, i.e. the proportion of email messages classified as spam that are truly spam:

$$SP = \frac{TP}{TP + FP} \quad (4.4)$$

- Ham (non-spam) Precision (HP): calculates the level to which emails not rejected by the spam filter and they are truly ham, i.e. the proportion of e-mails classified as ham that are truly ham:

$$HP = \frac{TN}{TN + FN} \quad (4.5)$$

- Spam Recall (SR): calculates the fraction of spam messages discarded by the spam filter, i.e. the proportion of spam e-mails that are classified as spam:

$$SR = \frac{TP}{Total\ Positive} = \frac{TP}{TP + FN} \quad (4.6)$$

- Ham (non-spam) Recall (HR): calculates the fraction of ham emails accepted by the filter, i.e. the proportion of ham e-mails that are classified as ham:

$$HR = \frac{TN}{Total\ Negative} = \frac{TN}{TN + FP} \quad (4.7)$$

- Spam F-Measure (FM): combined measure of the precision and recall metrics for spam class and it is calculated using the following formula:

$$FM (spam) = \frac{2 \times SP \times SR}{SP + SR} \quad (4.8)$$

- Ham F-Measure (FM): combined measure of the precision and recall metrics for ham class and it is calculated using the following formula:

$$FM (ham) = \frac{2 \times HP \times HR}{HP + HR} \quad (4.9)$$

- Receiver Operating Characteristics (ROC): is a curve showing the tradeoff between TPR and FPR. ROC is showing in a graphical diagram.
- Area under curve (AUC): measures the area under the ROC curve.

4.4 Spambase and SpamAssassin Experiment (using 3C-DCA)

We first applied the proposed DCA solutions using 3c-DCA on the Spambase dataset. The goal is to demonstrate the capability of DCA to filter the spam emails. There are three parameters in DCA: signals weights, number of DCA cells, and antigen multiplier. We selected the best values for the DCA parameters based on their impact on the performance in terms of AUC. We attempted different values for the number of cells and antigen multiplier in the range from 1 to 100. However, we only reported here the results of a sample of the tested values to manage the space.

First, we set the signal weights at their default values and changed the multiplier parameter as well as the number of DCs. The default parameter values used for DCA are shown in Table 15. Table 16 presents a sample of the results, where the best AUC occurs when the antigen multiplier is 100 and number of DCs is 30. Then, we fixed the above two parameters at their best values and changed the signal weights. The results of this series are shown in Table 17, where the best AUC is achieved at the highlighted values for the signal weights.

Table 15 DCA default parameters value

Parameters	Value
Number of cells	100
Antigen multiplier	50
Signals weights	$CSM = 2 \times PAMP + 1 \times Safe + 1 \times Danger$ $K = 2 \times PAMP - 3 \times Safe + 1 \times Danger$

Table 16: DCA AUC performance by changing Antigen multiplier and DCs

		Antigen Multiplier				
		10	30	50	70	100
No. of DCs	10	0.500	0.537	0.599	0.650	0.699
	30	0.514	0.649	0.727	0.759	0.761
	50	0.521	0.644	0.718	0.759	0.683
	70	0.524	0.619	0.699	0.723	0.761
	100	0.527	0.594	0.683	0.716	0.740

Table 17: DCA performance for several signals weights

$PAMP_{csm,w}$	$PAMP_{k,w}$	$Safe_{csm,w}$	$Safe_{k,w}$	$Danger_{csm,w}$	$Danger_{k,w}$	AUC
2	2	1	0	1	1	0.761
2	2	1	-3	1	1	0.999
2	1	1	-3	1	1	0.998
2	1	1	-3	10	1	0.986
15	1	0	-2	10	1	0.983
2	6	2	-1	1	1	0.968
2	2	1	-1	1	1	0.986
2	10	0	-1	1	1	0.95
2	6	2	0	1	1	0.761
2	6	2	-2	1	1	0.988
2	1	50	-3	40	1	0.979
3	10	0	-1	1	30	0.623
20	1	0	-2	10	1	0.983
1	10	0	-1	1	1	0.921
2	20	1	-3	1	1	0.973
3	10	0	-1	1	50	0.585

The results of the series of experiments show that DCA is very sensitive to several parameters. We achieved 0.999 AUC for spam classification in this case. The best case in this type of test was achieved by setting the DCA parameters as shown in Table 18. The MCAV values for a sample of antigens corresponding to the top 150 are shown in Figure 35. This diagram shows if the MCAV value of an antigen is greater than the threshold then it is a spam. The line in the diagram represents the threshold.

Table 18: Best DCA parameters values

Approach	Parameters	Values	AUC
3c-DCA	Signals weights	$CSM = 2 \times PAMP + 1 \times Safe + 1 \times Danger$ $K = 2 \times PAMP - 3 \times Safe + 1 \times Danger$	0.999
	Number of DCs	30	
	Antigen multiplier	100	

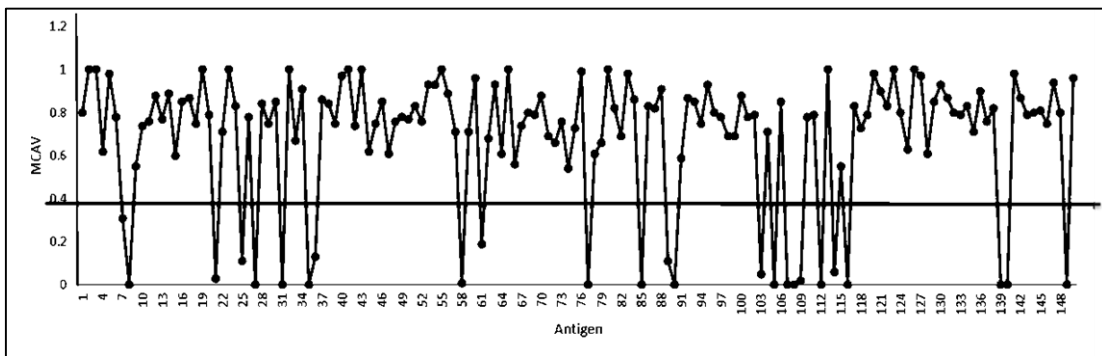


Figure 35: Sample of top 150 antigen MCAV values

We also compared the proposed models with KNN, Naïve Bayes, and SVM algorithms. Table 19 presents the performance measures of each classifier on Spambase dataset. Figure 36 shows the corresponding ROC for each classifier. It can be seen that both DCA-based models yield better results than the other classifiers.

We performed the same experiment on the SpamAssassin dataset and the results are illustrated in Table 20 and Figure 37. The results show that the 2C-DCA has better AUC performance.

Table 19: Performance for several classifiers on Spambase dataset

Classifier	Spam			Ham			Acc	AUC
	PRC	REC	F	PRC	REC	F		
KNN	0.773	0.780	0.776	0.856	0.851	0.853	0.823	0.815
Naïve Bayes	0.697	0.958	0.806	0.963	0.728	0.829	0.818	0.872
SVM	0.926	0.828	0.874	0.896	0.956	0.925	0.906	0.963
Majority Vote	0.883	0.905	0.894	0.937	0.922	0.929	0.915	0.957
3c-DCA	0.978	0.990	0.984	0.993	0.986	0.990	0.987	0.999
2c-DCA	0.964	0.996	0.979	0.997	0.976	0.986	0.983	0.999

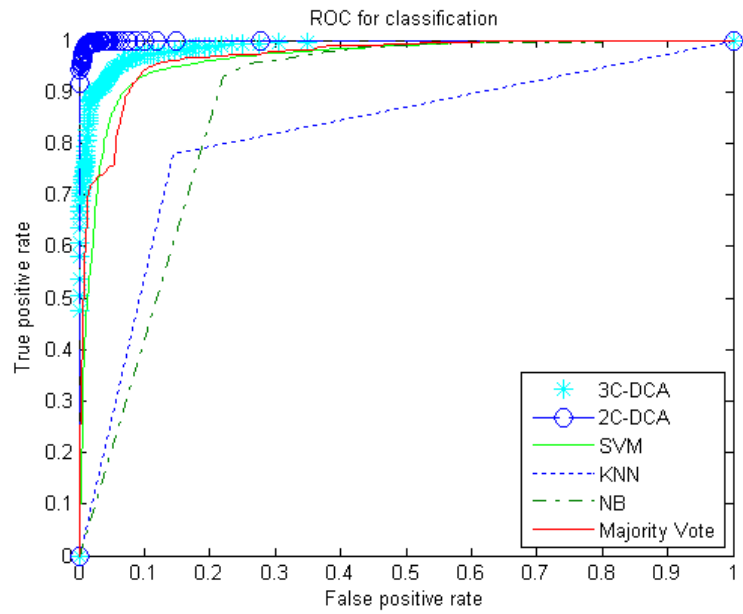


Figure 36: ROC performance comparison of various classifiers on Spambase dataset

Table 20: Performance for several classifiers on SpamAssassin dataset

Classifier	Spam			Ham			Acc	AUC
	PRC	REC	F	PRC	REC	F		
NB	0.920	0.470	0.621	0.658	0.961	0.781	0.722	0.854
SVM	0.961	0.529	0.681	0.689	0.981	0.809	0.762	0.917
KNN	0.693	0.616	0.652	0.672	0.741	0.704	0.680	0.678
Majority Vote	0.958	0.522	0.676	0.689	0.979	0.809	0.759	0.827
3C-DCA	0.969	0.971	0.970	0.990	0.989	0.989	0.984	0.955
2C-DCA	0.987	0.958	0.972	0.961	0.988	0.974	0.973	0.994

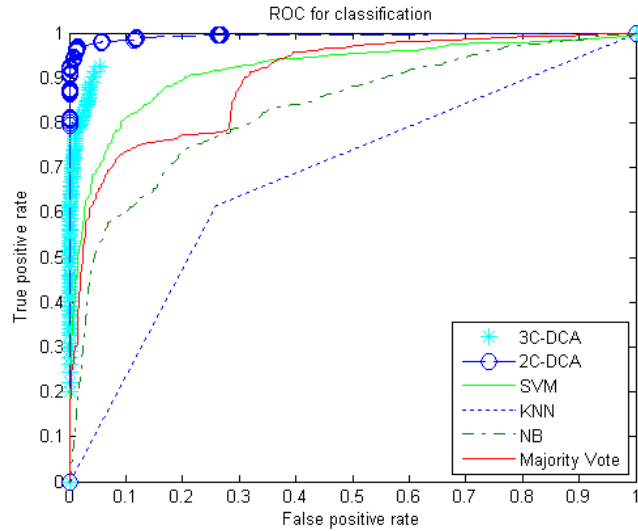


Figure 37: ROC performance comparison of various classifiers on SpamAssassin

4.5 Spam Experiments (using 2c-DCA)

This section describes 2c-DCA experiments. Moreover, the proposed features were leveraged in the experiments to see how effective these features are. We evaluated our model on four datasets: SMS big, SMS UCI, TREC2005, and SpamAssassin. We selected the best parameters based on AUC values.

Not every feature presents a good indicator. Some of the features perform well whereas others are weak features. Hence, we measured the relevance of each feature using the Gini Index (GI). GI calculates the weight of each feature with respect to the message classification. The higher the weight of a feature, the more relevant it is considered. The measured weights of the features are shown in Table 21- Table 24.

Based on the average results, SVM and NB perform better when all feature are selected as presented in Table 25 and Table 26. Therefore, we used all features in the DCA algorithm.

Table 21: Gini index weight SMS Big

Rank	Feature	Weight
1	Ind_350_Spam_Word	1
2	Ind_17k_Spam_Word	0.984339
3	Dep_Spam_Word	0.730443
4	Date Tag	0.211691
5	Function word	0.211471
6	Organization Tag	0.109447
7	Special characters	0.044159
8	Location Tag	0.029989
9	Price word (variant)	0.007501
10	Person Tag	0.005424
11	Gappy word	0.004595
12	Time Tag	0.003879
13	Money word (variant)	0.002265
14	Money Tags	0
15	Prescription word (variant)	0
16	Emotion Symbol	0

Table 22: Gini index weight SMS UCI

Rank	Feature	Weight
1	Ind_17k_Spam_Word	1
2	Ind_350_Spam_Word	0.979994
3	Dep_Spam_Word	0.72005
4	Special characters	0.248143
5	Date Tags	0.132973
6	Organization Tag	0.100215
7	Function word	0.057677
8	Location Tag	0.025524
9	Price word (variant)	0.018329
10	Gappy word	0.005612
11	Time Tag	0.004049
12	Money word (variant)	0.001331
13	Money Tags	7.98E-04
14	Person Tag	3.75E-04
15	Prescription word (variant)	4.05E-05
16	Emotion Symbol	0

Table 23: Gini index weight SpamAssassin

Rank	Feature	Weight
1	Ind_17k_Spam_Word in body	1
2	Ind_350_Spam_Word in body	0.951669
3	Spam domain	0.354979
4	Size of the message	0.327763
5	Number of users in “CC” fields	0.32356
6	Number of characters in “To”	0.321074
7	Ind_17k_Spam_Word in Subject	0.315864
8	Money word (variant)	0.297267
9	Ind_350_Spam_Word in Subject	0.295382
10	Subject Capital	0.199057
11	Number of users in “To”	0.15403
12	Dep_Spam_Word in Subject	0.092026
13	Function word	0.086715
14	Prescription word (variant)	0.05069
15	Dep_Spam_Word in body	0.046675
16	Special characters	0.035623
18	Blank “To” field	0.034407
19	Subject Length	0.03149
20	Attachment Count	0.026055
21	Viagra word (variant)	0.025204
22	Gappy word	0.001492
23	JavaScript	0
24	Cialis word (variant)	0

Table 24: Gini index weight TREC2005

Rank	Feature	Weight
1	Spam domain	1
2	Subject Length	0.374488
3	Dep_Spam_Word in body	0.200263
4	Special characters	0.11916
5	Number of users in “To”	0.119149
6	Number of characters in “To”	0.110908
7	Money word (variant)	0.106333
8	Size of the message	0.081175
9	Function word	0.076451
10	Ind_17k_Spam_Word in body	0.07319
11	Subject Capital	0.070441
12	Number of users in “CC” fields	0.064592
13	Prescription word (variant)	0.062367
14	Blank “To” field	0.051166
15	Ind_350_Spam_Word in body	0.034282
16	Viagra word (variant)	0.029446
18	Cialis word (variant)	0.024818
19	Gappy word	0.022766
20	Dep_Spam_Word in Subject	0.018308
21	Attachment Count	0.006854
22	Ind_350_Spam_Word in Subject	0.002628
23	Ind_17k_Spam_Word in Subject	0
24	JavaScript	0

Table 25: NB Classifier

Dataset	Top two Features								Top Five Features								All Features										
	Spam			Ham			Acc	AUC	Spam			Ham			Acc	AUC	Spam			Ham			Acc	AUC			
	PRC	REC	F	PRC	REC	F			PRC	REC	F	PRC	REC	F			PRC	REC	F	PRC	REC	F			PRC	REC	F
SMS BIG	0.901	0.945	0.922	0.982	0.966	0.974	0.959	0.988	0.859	0.952	0.902	0.984	0.948	0.965	0.948	0.988	0.786	0.955	0.861	0.984	0.915	0.948	0.924	0.973			
SMS UCI	0.734	0.902	0.809	0.984	0.949	0.966	0.943	0.973	0.785	0.901	0.838	0.984	0.962	0.973	0.954	0.966	0.602	0.926	0.729	0.988	0.904	0.944	0.907	0.961			
SpamAssassin	0.860	0.321	0.467	0.597	0.950	0.734	0.644	0.781	0.944	0.356	0.515	0.617	0.979	0.757	0.676	0.837	0.920	0.470	0.621	0.658	0.961	0.781	0.722	0.854			
TREC	1.000	0.411	0.578	0.666	1.000	0.799	0.734	0.870	0.952	0.537	0.683	0.714	0.977	0.825	0.780	0.919	0.678	0.978	0.799	0.969	0.594	0.732	0.774	0.930			
Average									0.903									0.928									0.930

Table 26: SVM Classifier

Dataset	Top two Features								Top Five Features								All Features										
	Spam			Ham			Acc	AUC	Spam			Ham			Acc	AUC	Spam			Ham			Acc	AUC			
	PRC	REC	F	PRC	REC	F			PRC	REC	F	PRC	REC	F			PRC	REC	F	PRC	REC	F			PRC	REC	F
SMS BIG	0.971	0.868	0.916	0.959	0.992	0.975	0.960	0.989	0.976	0.897	0.934	0.968	0.993	0.980	0.968	0.990	0.969	0.890	0.927	0.966	0.991	0.978	0.965	0.991			
SMS UCI	0.909	0.799	0.849	0.970	0.987	0.978	0.962	0.973	0.939	0.787	0.855	0.968	0.992	0.980	0.964	0.974	0.952	0.816	0.878	0.972	0.993	0.983	0.970	0.977			
SpamAssassin	0.816	0.543	0.651	0.672	0.883	0.763	0.718	0.799	0.956	0.550	0.697	0.697	0.976	0.813	0.769	0.884	0.961	0.529	0.681	0.689	0.981	0.809	0.762	0.917			
TREC	0.705	0.942	0.804	0.933	0.647	0.756	0.782	0.799	0.705	0.942	0.804	0.933	0.647	0.756	0.782	0.847	0.709	0.933	0.802	0.922	0.655	0.758	0.782	0.872			
Average									0.890									0.924									0.939

Table 27: KNN Classifier

Dataset	Top two Features								Top Five Features								All Features										
	Spam			Ham			Acc	AUC	Spam			Ham			Acc	AUC	Spam			Ham			Acc	AUC			
	PRC	REC	F	PRC	REC	F			PRC	REC	F	PRC	REC	F			PRC	REC	F	PRC	REC	F			PRC	REC	F
SMS BIG	0.669	0.984	0.795	0.994	0.840	0.910	0.875	0.824	0.908	0.887	0.896	0.964	0.970	0.967	0.949	0.860	0.926	0.894	0.908	0.966	0.976	0.971	0.955	0.872			
SMS UCI	0.705	0.827	0.759	0.973	0.945	0.959	0.930	0.782	0.867	0.824	0.845	0.973	0.980	0.977	0.959	0.808	0.881	0.851	0.865	0.977	0.982	0.980	0.964	0.499			
SpamAssassin	0.850	0.302	0.445	0.590	0.949	0.728	0.633	0.286	0.728	0.657	0.690	0.703	0.767	0.734	0.714	0.506	0.693	0.616	0.652	0.672	0.741	0.704	0.680	0.678			
TREC	0.461	1.000	0.631	0.000	0.000	0.000	0.452	0.500	0.696	0.859	0.765	0.853	0.666	0.740	0.756	0.571	0.712	0.649	0.677	0.723	0.775	0.746	0.720	0.714			
Average									0.598									0.686									0.691

4.5.1 SpamAssassin dataset

For SpamAssassin, the classification threshold was set to a MCAV of 0.487 (we used a sample of the dataset that consists of 1323 spam and 1390 non-spam). In the first series of experiments, we changed the multiplier parameter and number of DCs. Table 28 presents a sample of the AUC results of the DCA performance when changing the two parameters. This table shows that the best AUC when agent multiplier is 100 and number of DCs is 10.

Table 28: DCA AUC performance for SpamAssassin

		Antigen Multiplier				
		10	30	50	70	100
No. of DCs	10	0.8162	0.9287	0.9521	0.9573	0.9608
	30	0.7963	0.9571	0.959	0.9575	0.9593
	50	0.7417	0.9348	0.9598	0.9586	0.9588
	70	0.7136	0.9134	0.9521	0.9594	0.9579
	100	0.697	0.8955	0.944	0.9548	0.9576

Other experiments were then performed by changing the signals weights. The results of this series are shown in Table 29. The best AUC achieved result is highlighted. We achieved 0.994 AUC for spam classification in this case. The best case in this type of test was achieved by setting the DCA parameters as shown in

Table 30. We compared our proposed model with other machine-learning algorithms. It can be seen that our proposed model yields best AUC result when compared with other classifiers. The result of the comparison is shown in

Table 31 and Figure 38.

Table 29: DCA performance for several signals weight for SpamAssassin dataset

$PAMP_{csm,w}$	$PAMP_{k,w}$	$Safe_{csm,w}$	$Safe_{k,w}$	$Danger_{csm,w}$	$Danger_{k,w}$	AUC
2	2	1	0	1	1	0.961
2	2	1	-3	1	1	0.900
2	1	1	-3	1	1	0.838
2	1	1	-3	10	1	0.849
15	1	0	-2	10	1	0.783
2	6	2	-1	1	1	0.978
2	2	1	-1	1	1	0.986
2	3	1	-3	1	1	0.933
2	10	0	-1	1	1	0.994
2	6	2	0	1	1	0.955
2	6	2	-2	1	1	0.975
2	1	50	-3	40	1	0.913
3	10	0	-1	1	30	0.972
20	1	0	-2	10	1	0.783
1	10	0	-1	1	1	0.995
2	20	1	-3	1	1	0.989
3	10	0	-1	1	50	0.867

Table 30: Best DCA parameters value for SpamAssassin dataset

Parameters	Values
Signals weights	$CSM = 1 \times PAMP + 0 \times Safe + 1 \times Danger$ $K = 10 \times PAMP - 1 \times Safe + 1 \times Danger$
Number of DCs	10
Antigen multiplier	100

Table 31: Performance for several classifiers for SpamAssassin dataset

Classifier	Spam			Ham			Acc	AUC
	PRC	REC	F	PRC	REC	F		
NB	0.920	0.470	0.621	0.658	0.961	0.781	0.722	0.854
SVM	0.961	0.529	0.681	0.689	0.981	0.809	0.762	0.917
KNN	0.693	0.616	0.652	0.672	0.741	0.704	0.680	0.678
2C-DCA	0.987	0.958	0.972	0.961	0.988	0.974	0.973	0.994

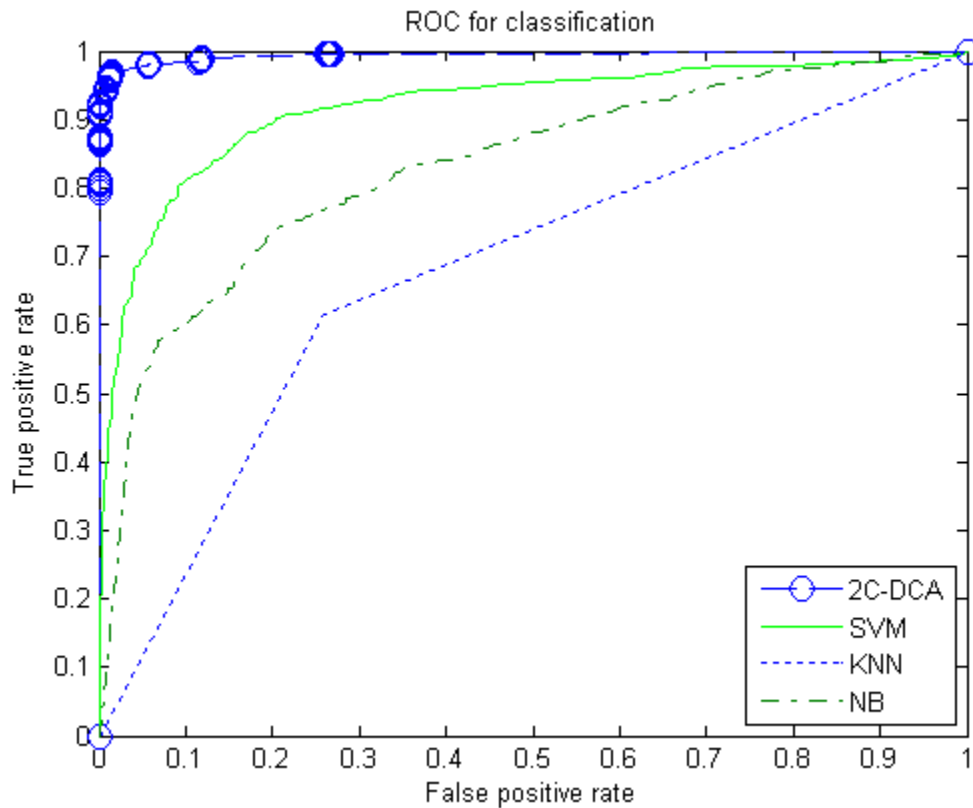


Figure 38: ROC for several classifiers for SpamAssassin dataset

4.5.2 TREC dataset

For TREC, the classification threshold was set to a MCAV of 0.452 (we used a sample of the dataset that consists of 228 spam and 276 non-spam). We performed the same series of experiments that were conducted in the previous section. We achieved 0.957 AUC for best parameters values. We compared our proposed model with other machine learning algorithms. The proposed model yields best AUC result when it is compared with other classifiers. The results are shown in Table 32 - Table 35 and the ROC is shown in Figure 39.

Table 32: DCA AUC performance TREC dataset

		Agent Multiplier				
		10	30	50	70	100
No. of DCs	10	0.554	0.663	0.721	0.761	0.784
	30	0.558	0.754	0.793	0.800	0.799
	50	0.563	0.720	0.786	0.799	0.799
	70	0.560	0.672	0.763	0.787	0.799
	100	0.551	0.654	0.754	0.776	0.795

Table 33: DCA performance for several signals weight for TREC dataset

$PAMP_{csm,w}$	$PAMP_{k,w}$	$Safe_{csm,w}$	$Safe_{k,w}$	$Danger_{csm,w}$	$Danger_{k,w}$	AUC
2	2	1	0	1	1	0.800
2	2	1	-3	1	1	0.944
2	1	1	-3	1	1	0.957
2	1	1	-3	10	1	0.938
15	1	0	-2	10	1	0.884
2	6	2	-1	1	1	0.865
2	2	1	-1	1	1	0.912
2	3	1	-3	1	1	0.939
2	10	0	-1	1	1	0.840
2	6	2	0	1	1	0.799
2	6	2	-2	1	1	0.891
2	1	50	-3	40	1	0.862
3	10	0	-1	1	30	0.790
20	1	0	-2	10	1	0.882
1	10	0	-1	1	1	0.784
2	20	1	-3	1	1	0.866
3	10	0	-1	1	50	0.790

Table 34: Best DCA parameters value for TREC dataset

Parameters	Values
Signals weights	$CSM = 2 \times PAMP + 1 \times Safe + 1 \times Danger$ $K = 1PAMP - 3 \times Safe + 1 \times Danger$
Number of DCs	30
Antigen multiplier	70

Table 35: Performance for several classifiers for TREC dataset

Classifier	Spam			Ham			Acc	AUC
	PRC	REC	F	PRC	REC	F		
NB	0.678	0.978	0.799	0.969	0.594	0.732	0.774	0.930
SVM	0.709	0.933	0.802	0.922	0.655	0.758	0.782	0.872
KNN	0.712	0.649	0.677	0.723	0.775	0.746	0.720	0.714
Proposed	0.793	0.974	0.874	0.973	0.790	0.872	0.873	0.957

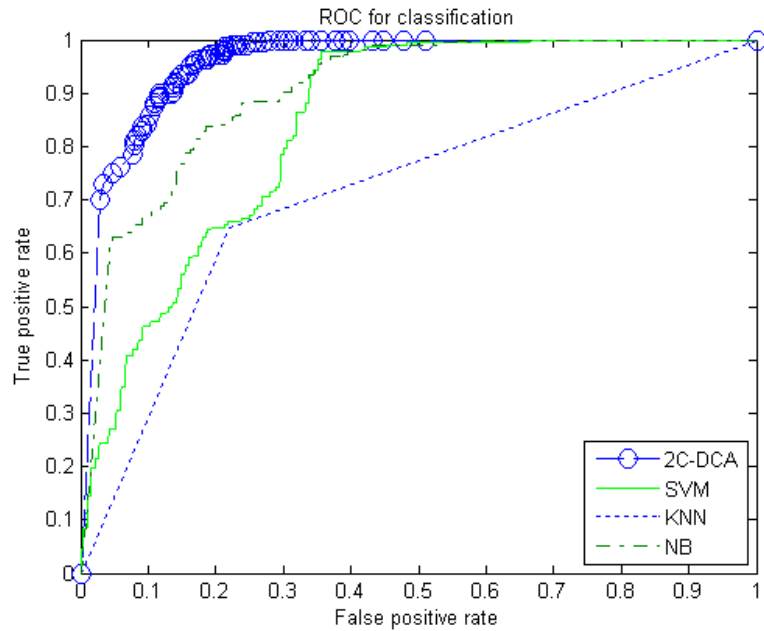


Figure 39: ROC for several classifiers for TREC dataset

4.5.3 SMS Big dataset:

For SMS Big, the classification threshold was set to a MCAV of 0.25 (we used sample of the dataset that consists of 320 spam and 960 non-spam). We performed the same series of experiments that were conducted in the previous sections. We achieved 0.999 AUC for best parameters values. We compared our proposed model with other machine-learning algorithms. Our proposed model yields best AUC result when it is compared with other classifiers. The result are shown in Table 36 - Table 39 and ROC is shown in Figure 40.

Table 36: DCA AUC performance for SMS Big dataset

		Agent Multiplier				
		10	30	50	70	100
No. of DCs	10	0.779	0.884	0.914	0.934	0.949
	30	0.782	0.930	0.952	0.954	0.955
	50	0.726	0.913	0.949	0.955	0.955
	70	0.716	0.903	0.942	0.950	0.955
	100	0.696	0.882	0.934	0.945	0.951

Table 37: DCA performance for several signals weight for SMS Big dataset

$PAMP_{csm,w}$	$PAMP_{k,w}$	$Safe_{csm,w}$	$Safe_{k,w}$	$Danger_{csm,w}$	$Danger_{k,w}$	AUC
2	2	1	0	1	1	0.955
2	2	1	-3	1	1	1.000
2	1	1	-3	1	1	1.000
2	1	1	-3	10	1	0.995
15	1	0	-2	10	1	0.974
2	6	2	-1	1	1	0.998
2	2	1	-1	1	1	1.000
2	3	1	-3	1	1	1.000
2	10	0	-1	1	1	0.995
2	6	2	0	1	1	0.955
2	6	2	-2	1	1	1.000
2	1	50	-3	40	1	0.990
3	10	0	-1	1	30	0.932
20	1	0	-2	10	1	0.973
1	10	0	-1	1	1	0.990
2	20	1	-3	1	1	0.999
3	10	0	-1	1	50	0.896

Table 38: Best DCA parameters value for SMS Big dataset

Parameters	Values
Signals weights	$CSM = 2 \times PAMP + 1 \times Safe + 1 \times Danger$ $K = 2 \times PAMP - 3 \times Safe + 1 \times Danger$
Number of DCs	50
Antigen multiplier	70

Table 39: Performance for several classifiers for SMS Big dataset

Classifier	Spam			Ham			Acc	AUC
	PRC	REC	F	PRC	REC	F		
NB	0.786	0.955	0.861	0.984	0.915	0.948	0.924	0.973
SVM	0.969	0.890	0.927	0.966	0.991	0.978	0.965	0.991
KNN	0.926	0.894	0.908	0.966	0.976	0.971	0.955	0.872
Proposed	0.973	1.000	0.986	1.000	0.991	0.995	0.993	0.999

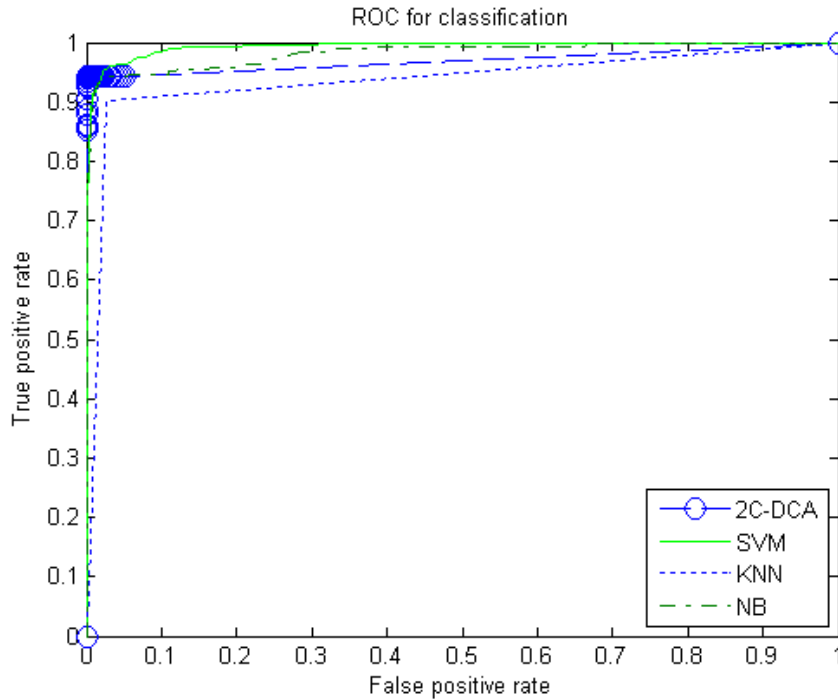


Figure 40: ROC for several classifiers for SMS Big dataset

4.5.4 SMS UCI dataset:

For SMS UCI, the classification threshold was set to a MCAV of 0.134 (we used 747 spam and 4827 non-spam). We performed the same series of experiments that were conducted in the previous sections. We achieved 99.9 % AUC for best parameters values. We compared our proposed model with other machine-learning algorithms. Our proposed model yields best AUC result when it is compared with other classifiers. The results are shown in Table 40 - Table 43. The ROC is shown in Figure 41.

Table 40: DCA AUC performance for SMS UCI dataset

		Agent Multiplier				
		10	30	50	70	100
No. of DCs	10	0.672	0.858	0.909	0.933	0.946
	30	0.699	0.931	0.948	0.948	0.948
	50	0.655	0.901	0.947	0.948	0.948
	70	0.636	0.864	0.935	0.948	0.948
	100	0.617	0.821	0.932	0.940	0.947

Table 41: DCA performance for several signals weight for SMS UCI dataset

PAMP _{csm,w}	PAMP _{k,w}	Safe _{csm,w}	Safe _{k,w}	Danger _{csm,w}	Danger _{k,w}	AUC
2	2	1	0	1	1	0.948417
2	2	1	-3	1	1	0.999999
2	1	1	-3	1	1	0.999761
2	1	1	-3	10	1	0.992381
15	1	0	-2	10	1	0.961558
2	6	2	-1	1	1	0.999422
2	2	1	-1	1	1	0.999999
2	3	1	-3	1	1	0.999999
2	10	0	-1	1	1	0.999945
2	6	2	0	1	1	0.948409
2	6	2	-2	1	1	0.999902
2	1	50	-3	40	1	0.990025
3	10	0	-1	1	30	0.896799
20	1	0	-2	10	1	0.961498
1	10	0	-1	1	1	0.999712
2	20	1	-3	1	1	0.999940
3	10	0	-1	1	50	0.765086

Table 42: Best DCA parameters value for SMS UCI dataset

Parameters	Values
Signals weights	$CSM = 2 \times PAMP + 1 \times Safe + 1 \times Danger$ $K = 2 \times PAMP - 3 \times Safe + 1 \times Danger$
Number of DCs	30
Antigen multiplier	100

Table 43: Performance for several classifiers for SMS UCI dataset

Classifier	Spam			Ham			Acc	AUC
	PRC	REC	F	PRC	REC	F		
NB	0.602	0.926	0.729	0.988	0.904	0.944	0.907	0.961
SVM	0.952	0.816	0.878	0.972	0.993	0.983	0.970	0.977
KNN	0.881	0.851	0.865	0.977	0.982	0.980	0.964	0.903
Proposed	0.959	1.000	0.979	1.000	0.993	0.997	0.994	0.999

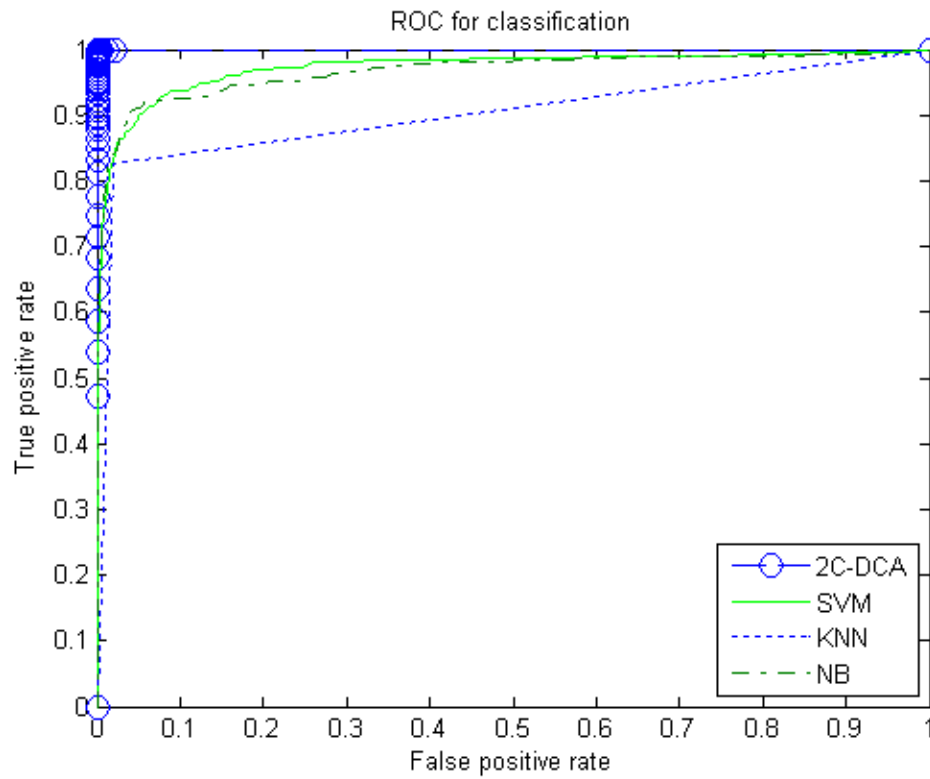


Figure 41: ROC for several classifiers for SMS UCI dataset

4.6 Email Message Multi-Category Classification

This section explains how to use DCA to solve the multi-category classification problem of email messages.

The DCA algorithm was designed to solve binary classification problems, i.e. it classifies objects into two classes only. To utilize DCA for multi-category classification and achieve our objective, we used one-versus-all (also known as one-vs-rest) approach [71]. In this approach, the binary classifier is constructed to separate one class from other classes as shown in Figure 42. For N classes, N binary classifiers are constructed and each binary classifier classifies one class as positive versus other classes as negative [84].

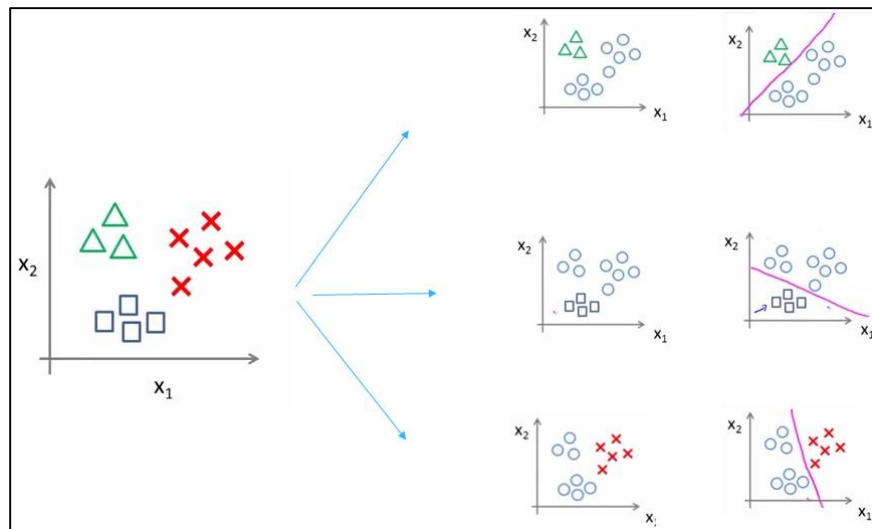


Figure 42: One-versus-all²⁹

²⁹ coursera.org (Machine Learning course by Stanford University)

For Email Message Multi-Category classification experiments, we used Enron dataset. As part of the dataset preprocessing, we dropped irrelevant/non-topical subfolders such as “deleted”, “inbox”, “trash”, and “sent”. Moreover, subcategories with a few/very large number of emails were dropped to have a balanced dataset. The number of folders (classes) for each user (dataset) is presented in Table 44. The results of our experiments produced accuracy more than 90% in most of the datasets as shown in Table 45 and Figure 43.

Table 44: Users datasets details

User	No. of subfolders (classes)	No. of features	Total number of email messages
arnold-j	5	624	68
blair-l	6	914	101
cash-m	6	2201	108
williams-w3	7	514	108
sanders-r	8	1310	158
shackleton-s	8	744	120
lokay-m	9	3859	1328
farmer-d	9	1520	235
beck-s	13	908	154
shapiro-r	13	1885	214

Table 45: Comparison of percentage classification accuracy per user

No.	User	KNN	NB	SVM	Proposed
1	arnold-j	0.985	0.986	1.00	1.00
2	blair-l	0.486	0.61	0.671	0.584
3	cash-m	0.798	0.739	0.825	0.917
4	williams-w3	0.860	0.724	0.907	0.907
5	sanders-r	0.975	0.929	0.975	0.975
6	shackleton-s	0.842	0.825	0.892	0.921
7	lokay-m	0.363	0.669	0.202	0.519
8	farmer-d	0.829	0.766	0.831	0.916
9	beck-s	0.793	0.753	0.838	0.922
10	shapiro-r	0.776	0.739	0.847	0.963

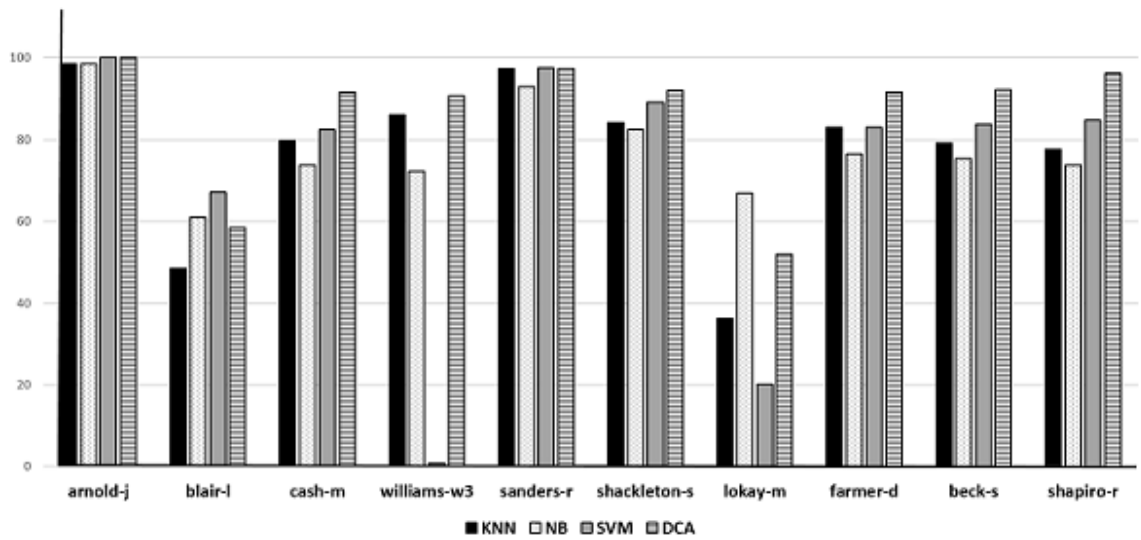


Figure 43: Comparison of classification accuracy per user

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

With the evolution of technology, hackers are leveraging every possible way to compromise the end-user's systems including his/her mobile and computer. Email and SMS spam is widely spread. Cybercrimes have targeted end-user email over the past few years but recently they start also targeting mobile devices. The increase of hacking attempts against mobile phone is because of the increase in the number of mobile of devices and amount of data that these devices are containing. Spam becomes a serious security issue. It annoys the end user and wastes computing and organization resources. To overcome these issues, several spam filtering solutions have been proposed in the literature. The anti-spam has become an important control to mitigate the risk.

This research gives a better understanding of text spam email and spam SMS filtration methods. In addition, it discusses email text multi-category classification (also known as email sorting). It explains the general architecture of the spam classifier and its main components and phases in a holistic approach. We also, explain the impact of extracting the right features from the email or spam to significantly enhance the performance of the classifiers. An overview of different performance measures were provided to help selecting the best approach and parameters' values.

This research investigates the use of dendritic cell algorithm (DCA), which was developed based on the reaction of the human immune system to an intruder. We provided an explanation of the algorithm and its application to solve several information security issues. We developed two models based on DCA algorithm to filter spam. The first model uses three classifiers (3c-DCA) to generate the required signal to operate the DCA algorithm. The second model, requires two classifiers (2c-DCA) to generate the signals and it needs less computational resources. We evaluated our model using AUC which is widely used in the literature to measure the overall classifier performance. The results show that AUC and accuracy of the proposed model are outstanding. The proposed model is effective and can be easily integrated with other existing solutions that depend on machine learning algorithms to enhance the accuracy of the anti-spam solution. The proposed model was compared with widely used machine learning algorithms such as SVM, KNN and Naïve Bayes. The model was benchmarked on several email and SMS spam datasets.

In addition, we proposed using several features that have been evaluated using several machine learning algorithms and the results yield a significant enhancement on the classifiers.

We also applied the model to solve multi-classification problem (also known as email sorting). One-versus-all (also known as one-vs-rest) approach were used to construct the multi-classification model.

The empirical results showed that our proposed model produced remarkable results. We were able to achieve AUC close to 100% in most the experiments.

5.2 Future work

The use of DCA to solve spam filtration and multi-category classification problem has not received a lot of attention in the literature. As a future work, one can:

1. Explore using other feature extraction approaches to reduce the processing time.
2. Investigate an automated approach to find the optimum values of DCA parameters to make the DCA a dynamic algorithm. Currently, the trial and error approach is used to identify the values of the best DCA parameters. The trial and error values are not easy to figure out because of the number of parameters and space.
3. Expand the model to include spam emails with images.

References

- [1] “Email Statistics Report 2015-2019,” The Radicati Group, Mar. 2015 [Online]. Available: <http://www.radicati.com/wp/wp-content/uploads/2015/02/Email-Statistics-Report-2015-2019-Executive-Summary.pdf>
- [2] G. V. Cormack and T. R. Lynam, “Spam Corpus Creation for TREC,” in *Proceedings of the Second Conference on Email and Anti-Spam (CEAS)*, 2005, pp. 162–163.
- [3] “McAfee Labs Threats Report,” McAfee, Feb. 2015 [Online]. Available: <http://www.mcafee.com/us/resources/reports/rp-quarterly-threat-q4-2014.pdf>
- [4] “Symantec Intelligence Report,” Feb. 2015 [Online]. Available: http://www.symantec.com/content/en/us/enterprise/other_resources/b-intelligence-report-02-2015-en-us.pdf
- [5] R. Jennings, “Cost of Spam is Flattening — Our 2009 Predictions,” Jan. 2009 [Online]. Available: <http://email-museum.com/2009/01/28/cost-of-spam-is-flattening-our-2009-predictions/>
- [6] E. Blanzieri and A. Bryl, “A Survey of Learning-Based Techniques of Email Spam Filtering,” *Artificial Intelligence Review*, vol. 29, no. 1, pp. 63–92, 2008.
- [7] L. H. Gomes, C. Cazita, J. M. Almeida, V. Almeida, and W. Meira Jr, “Characterizing a Spam Traffic,” in *Proceedings of the ACM Internet Measurement Conference*, 2004, pp. 356–369.
- [8] S. Firake, P. Soni, and B. B. Meshram, “Phishing E-mail Analysis,” *International Journal of Computer Science & Emerging Technologies*, vol. 2, no. 1, Feb. 2011.
- [9] B. Medlock, “An Adaptive, Semi-Structured Language Model Approach to Spam Filtering on a New Corpus,” in *Proceedings of the Third Conference on Email and Anti-Spam*, 2006.
- [10] G. V. Cormack, J. M. Gómez Hidalgo, and E. P. Sánchez, “Spam Filtering for Short Messages,” in *Proceedings of the 16th ACM Conference on Information and Knowledge Management*, 2007, pp. 313–320.

- [11] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami, “Contributions to the Study of SMS Spam Filtering: New Collection and Results,” in *Proceedings of the 11th ACM Symposium on Document Engineering*, 2011, pp. 259–262.
- [12] J. M. Gómez Hidalgo, M. M. López, and E. P. Sanz, “Combining Text and Heuristics for Cost-Sensitive Spam Filtering,” in *Proceedings of the 4th Computational Natural language Learning Workshop*, Lisbon, Portugal, 2000, pp. 99–102.
- [13] H. Wang, Y. Yu, and Z. Liu, “SVM Classifier Incorporating Feature Selection Using GA for Spam Detection,” in *Embedded and Ubiquitous Computing*, Springer, 2005, pp. 1147–1154.
- [14] Y. Yang and S. Elfayoumy, “Anti-Spam Filtering Using Neural Networks and Bayesian Classifiers,” in *Proceedings of the International Symposium on Computational Intelligence in Robotics and Automation*, 2007, pp. 272–278.
- [15] E.-S. El-Alfy, “Learning Methods for Spam Filtering,” in *Computer Systems, Support and Technology*, Nova Science Publishers, 2011.
- [16] W. Zhao and Z. Zhang, “An Email Classification Model Based on Rough Set Theory,” in *Active Media Technology*, 2005, pp. 403–408.
- [17] C. S. Oliveira, F. G. Cozman, and I. Cohen, “Splitting the Unsupervised and Supervised Components of Semi-Supervised Learning,” in *Proceedings of the 22nd International Conference on Machine Learning Workshop*, 2005, pp. 67–73.
- [18] C. Perlich, F. Provost, and J. S. Simonoff, “Tree Induction vs. Logistic Regression: a Learning-curve Analysis,” *The Journal of Machine Learning Research*, vol. 4, pp. 211–255, 2003.
- [19] D. M. Tax and C. J. Veenman, “Turning the Hyperparameter of an AUC-Optimized Classifier,” in *Proceedings of the Seventeenth Belgium-Netherlands Conference on Artificial Intelligence*, 2005, pp. 224–231.
- [20] L. Zhang, J. Zhu, and T. Yao, “An Evaluation of Statistical Spam Filtering Techniques,” *ACM Transactions on Asian Language Information Processing (TALIP)*, vol. 3, no. 4, pp. 243–269, 2004.

- [21] I. Androutsopoulos, “Learning to Filter Unsolicited Commercial E-Mail,” Technical report 2004/2, NCSR “Demokritos,” 2004 [Online]. Available: http://www.aueb.gr/users/ion/docs/TR2004_updated.pdf
- [22] A. Bratko, B. Filipič, G. V. Cormack, T. R. Lynam, and B. Zupan, “Spam Filtering Using Statistical Data Compression Models,” *The Journal of Machine Learning Research*, vol. 7, pp. 2673–2698, 2006.
- [23] K. N. Junejo, M. M. Yousaf, and A. Karim, “A Two-pass Statistical Approach for Automatic Personalized Spam Filtering,” in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, Discovery Challenge Workshop, 2006, pp. 16–27.
- [24] S. Bickel, “ECML-PKDD Discovery Challenge 2006 Overview,” *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD) Discovery Challenge*, 2006.
- [25] Z. Yang, X. Nie, W. Xu, and J. Guo, “An Approach to Spam Detection by Naive Bayes Ensemble Based on Decision Induction,” in *Intelligent Systems Design and Applications*, 2006, vol. 2, pp. 861–866.
- [26] P. Bermejo, J. A. Gamez, J. M. Puerta, and R. Uribe-Paredes, “Improving KNN-based E-Mail Classification into Folders Generating Class-Balanced Datasets,” in *Proceedings of the 12th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 2008, pp. 529–536.
- [27] A. A. Alsallab, M. A. Rashwan, “E-Mail Classification Using Deep Networks,” *Journal of Theoretical and Applied Information Technology*, vol. 37, no. 2, pp. 242–251, Mar. 2012.
- [28] G. Cormack and T. Lynam, “TREC 2005 Spam Track Overview,” in *Proceedings of the Fourteenth Text REtrieval Conference*, 2005.
- [29] J. Goodman and W. Yih, “Online Discriminative Spam Filter Training,” in *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS)*, 2006.

- [30] G. Cormack, “TREC 2006 Spam Track Overview,” in *Proceedings of Text REtrieval Conference (TREC)*, 2006.
- [31] G. V. Cormack, “TREC 2007 spam track overview,” in *Proceedings of the 6th Text REtrieval Conference (TREC)*, 2007.
- [32] L. Zhang, J. Zhu, and T. Yao, “An Evaluation of Statistical Spam Filtering Techniques,” *ACM Transactions on Asian Language Information Processing (TALIP)*, vol. 3, no. 4, pp. 243–269, 2004.
- [33] J. M. Gómez Hidalgo, G. C. Bringas, E. P. Sáenz, and F. C. García, “Content Based SMS Spam Filtering,” in *Proceedings of the 2006 ACM symposium on Document engineering*, 2006, pp. 107–114.
- [34] G. V. Cormack, J. M. G. Hidalgo, and E. P. Sáenz, “Feature Engineering for Mobile (SMS) Spam Filtering,” in *Proceedings of the 30th Annual International ACM Special Interest Group on Information Retrieval (SIGIR)*, 2007, pp. 871–872.
- [35] I. Fette, N. Sadeh, and A. Tomasic, “Learning to Detect Phishing Emails,” in *Proceedings of the 16th International Conference on World Wide Web*, 2007, pp. 649–656.
- [36] M. Chandrasekaran, K. Narayanan, and S. Upadhyaya, “Phishing Email Detection Based on Structural Properties,” in *Proceedings of New York State (NYS) Cyber Security Conference*, 2006, pp. 1–7.
- [37] I. R. A. Hamid and J. Abawajy, “Hybrid Feature Selection for Phishing Email Detection,” in *Algorithms and Architectures for Parallel Processing*, Springer, 2011, pp. 266–275.
- [38] Q. Ren, “Feature Fusion Framework for Spam Filtering Based on SVM,” in *Proceedings of the 7th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference*, 2010, pp. 1–6.
- [39] C.-C. Lai and M.-C. Tsai, “An Empirical Performance Comparison of Machine Learning Methods for Spam E-mail Categorization,” in *Proceedings of the 4th International Conference Hybrid Intelligent Systems*, 2004, pp. 44–48.

- [40] H. Drucker, S. Wu, and V. N. Vapnik, "Support Vector machines for Spam Categorization," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1048–1054, 1999.
- [41] M. Bazarganigilani, "Phishing E-Mail Detection Using Ontology Concept and Naïve Bayes Algorithm," *International Journal of Research and Reviews in Computer Science*, vol. 2, no. 2, pp. 249–252, Apr. 2011.
- [42] W. N. Gansterer and D. Pölz, "E-mail Classification for Phishing Defense," in *Advances in Information Retrieval*, Springer, 2009, pp. 449–460.
- [43] S. Maldonado and R. Weber, "Embedded Feature selection for Support Vector Machines: State of the Art and Future Challenges," in *Pattern Recognition, Image Analysis, Computer Vision, and Applications*, Springer, 2011, pp. 304–311.
- [44] M. Blachnik, "Comparison of Various Feature Selection Methods in Application to Prototype Best Rules," in *Computer Recognition Systems*, Springer, 2009, pp. 257–264.
- [45] P. Cortez, R. F. M. Vaz, M. Rocha, M. Rio, and P. Sousa, "Evolutionary Symbiotic Feature Selection for Email Spam Detection," *SciTePress*, Jul. 2012.
- [46] L. Cao, N. Guihua, and L. Pingfeng, "Ontology-Based Spam Detection Filtering System," in *Proceedings of the International Conference on Business Management and Electronic Information (BMEI)*, Guangzhou, 2011, vol. 3, pp. 282–284.
- [47] D. Kumar and P. Rana, "Design and Development of a Stemmer for Punjabi," *International Journal of Computer Applications*, 2011.
- [48] J. Provost, "Naive-Bayes vs. Rule-Learning in Classification of Email," Artificial Intelligence Lab, University of Texas at Austin, Technical Report, AI-TR-99-284, 1999 [Online]. Available: <ftp://ftp.cs.utexas.edu/pub/AI-Lab/tech-reports/UT-AI-TR-99-284.pdf>.
- [49] C.-H. Wu, "Behavior-Based Spam Detection Using a Hybrid Method of Rule-Based Techniques and Neural Networks," *Expert Systems with Applications*, vol. 36, no. 3, pp. 4321–4330, 2009.

- [50] K. Tretyakov, "Machine Learning Techniques in Spam Filtering," Institute of Computer Science, University of Tartu Data Mining Problem-oriented Seminar, MTAT, 2004, vol. 3, pp. 60–79.
- [51] X. Carreras and L. Marquez, "Boosting Trees for Anti-spam Email Filtering," in *Proceedings of the 4th International Conference on Recent Advances in Natural Language Processing*, Bulgaria 2001.
- [52] W. Shang, H. Huang, H. Zhu, Y. Lin, Y. Qu, and Z. Wang, "A novel feature selection algorithm for text categorization," *The Journal of Expert System with Applications*, 33:1-5, 2007.
- [53] B. Leiba, J. Ossher, V. T. Rajan, R. Segal, and M. N. Wegman, "Method for Recognizing Spam Email," U.S. Patent 7475118, Jan. 2009.
- [54] H.-Y. Lam and D.-Y. Yeung, "A Learning Approach to Spam Detection Based on Social Networks," in *Proceedings of the 4th Conference on Email and Anti-Spam (CEAS)*, 2007.
- [55] D. Dasgupta, S. Yu, and F. Nino, "Recent Advances in Artificial Immune Systems: Models and Applications," *Applied Soft Computing, Elsevier*, vol. 11, no. 2, pp. 1574–1587, 2011.
- [56] M. Gendreau and J.-Y. Potvin, *Handbook of Metaheuristics*, vol. 2. Springer, 2010.
- [57] J. Timmis, M. Neal, and J. Hunt, "An Artificial Immune System for Data Analysis," *Biosystems*, vol. 55, no. 1, pp. 143–150, 2000.
- [58] J. Greensmith and U. Aickelin, "The Deterministic Dendritic Cell Algorithm," in *7th ICARIS*, 2008, vol. LNCS 5132, pp. 291–302.
- [59] J. Greensmith, J. Feyereisl, and U. Aickelin, "The DCA: Some Comparison," *Evolutionary Intelligence, Springer*, vol. 1, no. 2, pp. 85–112, 2008.
- [60] J. Greensmith and U. Aickelin, "The Dendritic Cell Algorithm," PhD Thesis, University of Nottingham, Nottingham, UK, 2007.

- [61] J. Greensmith, U. Aickelin, and G. Tedesco, "Information Fusion for Anomaly Detection with the Dendritic Cell Algorithm," *Special Issue on Biologically-Inspired Information Fusion*, vol. 11, no. 1, pp. 21–34, Jan. 2010.
- [62] E. Bendiab and M. K. Kholadi, "Recognition of Plant Leaves Using the Dendritic Cell Algorithm," *International Journal of Digital Information and Wireless Communications (IJDIWC)*, vol. 1, no. 1, pp. 284–292, 2011.
- [63] R. Huang, H. Tawfik, and A. K. Nagar, "Artificial Dendritic Cells Algorithm for Online Break-in Fraud Detection," in *Proceedings of the 2nd International Conference on Developments in eSystems Engineering (DESE)*, 2009, pp. 181–189.
- [64] S. J. Delany, M. Buckley, and D. Greene, "SMS Spam Filtering: Methods and Data," *Expert Systems with Applications*, vol. 39, no. 10, pp. 9899–9908, Aug. 2012.
- [65] Kevin Gao, "A List of Common Spam Words," Feb-2014 [Online]. Available: <http://emailmarketing.com100.com/email-marketing-ebook/spam-words.aspx>
- [66] S. Argamon and S. Levitan, "Measuring the Usefulness of Function Words for Authorship Attribution," In *Proceedings of the Joint Conference of the Association for Computers and the Humanities and the Association for Literary and Linguistic Computing (ACH/ALLC)*, Victoria, BC, Canada, 2005.
- [67] Sequence Publishing, "Function Words." [Online]. Available: <http://www.sequencepublishing.com/academic.html>. [Accessed: 21-Sep-2014].
- [68] D. A. Hull, "Stemming Algorithms: A Case Study for Detailed Evaluation," *Journal of the American Society for Information Science (JASIS)*, vol. 47, no. 1, pp. 70–84, 1996.
- [69] C. C. Aggarwal and C. Zhai, "A Survey of Text Classification Algorithms," in *Mining Text Data*, Springer, 2012, pp. 163–222.
- [70] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, 1st edition. New York: Cambridge University Press, 2008.
- [71] K.-B. Duan, J. C. Rajapakse, and M. N. Nguyen, "One-versus-One and One-versus-All Multiclass SVM-RFE for Gene Selection in Cancer Classification," in

Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, Springer, 2007, pp. 47–56.

- [72] G. Caruana and M. Li, “A Survey of Emerging Approaches to Spam Filtering,” *ACM Computing Surveys*, vol. 44, no. 2, pp. 1–27, 2012.
- [73] D.-H. Shih, H.-S. Chiang, and B. Lin, “Collaborative Spam Filtering with Heterogeneous Agents,” *Expert Systems with Applications*, vol. 35, no. 4, pp. 1555–1566, 2008.
- [74] A. Gray and M. Haahr, “Personalised, Collaborative Spam Filtering,” in *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, CA, USA, July-August, 2004.
- [75] X. Fang and J. Liu, “Input Data Preprocessing for Bots Detection Using the Dendritic Cells Algorithm,” in *Proceedings of the 6th International Congress on Image and Signal Processing (CISP)*, 2013, vol. 03, pp. 1362–1366.
- [76] Tom Mitchell, *Machine Learning*, McGraw Hill, 1997.
- [77] C. Aggarwal and C. Zhai, *Mining Text Data*, Springer, 2012.
- [78] R. Kohavi and G. John, “Wrappers for Feature Subset Selection,” *Artificial Intelligence*, Special Issue on Relevance, vol. 97, no. 1–2, pp.273–324, 1996.
- [79] M. Marsono, “*Towards Improving E-mail Content Classification for Spam Control: Architecture, Abstraction, and Strategies*” PhD Thesis, University of Victoria, 2007.
- [80] M. Wong and W. Schlitt, “*Sender Policy Framework (SPF) for Authorizing Use of Domains in E-mail*,” RFC 4408, 2006.
- [81] Gu F, Greensmith J, Aicklein U, “*The Dendritic Cell Algorithm for Intrusion Detection*,” In *Biologically Inspired Networking and Sensing: Algorithms and Architectures*, IGI Global, pp. 84-102, 2011.
- [82] E.-S. El-Alfy and R. E. Abdel-Aal, “Spam Filtering with Abductive Networks,” in *IEEE World Congress on Computational Intelligence*, 2008, pp. 165–170.
- [83] C. Cortes and V. Vapnik, “Support-Vector Networks,” *Machine Learning*, vol. 20, no.3, pp. 273–297, 1995.

Appendix I. Stop Words

Stop words are words that are so common and often filtered out during preprocessing. But unfortunately, there is no definitive list of them. Hence, in our experiments, we used the following 395 stop words from³⁰:

a	each	needn	themselves	close
abaft	early	needs	then	concerning
aboard	either	neither	there	considering
about	em	never	these	cos
above	english	nevertheless	they	could
across	enough	new	thine	couldn
afore	ere	next	this	couldst
aforesaid	even	nigh	tho ³¹	dare
after	ever	nigher	those	dared
again	every	nighest	thou	daren
against	everybody	nisi	though	dares
agin ³²	everyone	no	three	daring
ago	everything	one	thro	despite
aint ³³	except	nobody	through	did
albeit	excepting	none	throughout	didn
all	failing	nor	thru	different
almost	far	not	thyself	directly
alone	few	nothing	till	do
along	first	notwithstanding	to	does
alongside	five	now	today	doesn

³⁰ <http://rapidminernotes.blogspot.com/2015/01/english-stop-words.html>

³¹ slang word for "though"

³² dialect form of against.

³³ am not; are not; is not.

already	following	er	together	doing
also	for	of	too	done
although	four	off	touching	don
always	from	often	toward	dost
am	gonna ³⁴	on	towards	doth
american	gotta ³⁵	once	TRUE	down
amid	had	oneself	twas	during
amidst	hadn	only	tween	durst
among	hard	onto	twere	let
amongst	has	open	twill	like
an	hasn	or	twixt	likewise
and	hast	other	two	little
anent	hath	otherwise	twould	living
another	have	ought	under	long
any	haven	oughtn	underneath	many
anybody	having	our	unless	may
anyone	he	ours	unlike	mayn
anything	her	ourselves	until	me
are	here	out	unto	mid
aren	hers	outside	up	midst
around	herself	over	upon	might
as	high	own	us	mightn
aslant	him	past	used	mine
astride	himself	pending	usually	minus
at	his	per	versus	more
athwart	home	perhaps	very	most
away	how	plus	via	much

³⁴ going to.

³⁵ go to.

back	howbeit	possible	vice	must
bar	however	present	vis-a-vis	mustn
barring	id	probably	wanna	my
be	if	provided	wanting	myself
because	ill	providing	was	near
been	immediately	public	wasn	neath ³⁶
before	important	qua	way	need
behind	in	quite	we	needed
being	inside	rather	well	needing
below	instantly	re	were	shed
beneath	into	real	weren	shell
beside	is	really	wert ³⁷	short
besides	isn	respecting	what	should
best	it	right	whatever	shouldn
better	its	round	when	since
between	itself	same	whencesoever	six
betwixt	ve	sans	whenever	small
beyond	just	save	whereas	so
both	large	saving	where	some
but	last	second	whether	somebody
by	later	several	which	someone
can	least	shall	whichever	something
cannot	left	shalt	whichsoever	sometimes
certain	less	shan	while	soon
circa	lest	she	whilst	special
who	whosoever	wouldst	yourselves	that
whoever	will	ye	still	the

³⁶ beneath.

³⁷ the imperfect subjunctive of "were" found in Shakespearean English.

whole	with	yet	such	thee
whom	within	you	summat ³⁸	their
whore	without	your	supposing	theirs
whose	wont	yours	sure	them
whoso	would	yourself	than	wouldn

³⁸ Yorkshire dialect for "something"

Vitae

Name : Ali A. AlHasan

Nationality : Saudi

Date of Birth :2/6/1983

Email : ALI.HASAN.5@OUTLOOK.COM

Address : Dhahran, Saudi Arabia

Academic Background : B.S. IN COMPUTER SCIENCE, KFUPM, 2006

Professional Certifications : CISSP, CISA, CISM, GCIH, COBIT, ITIL, GWAPT

Area of Interest : Information Security and Data Mining

Publications:

- E.-S. M. El-Alfy and A. A. Al-Hasan, “A novel bio-inspired predictive model for spam filtering based on dendritic cell algorithm,” in *Proceedings of the IEEE Symposium on Computational Intelligence in Cyber Security (CICS)*, Dec. 2014.
- A. A. Al-Hasan and E.-S.M. El-Alfy, “Dendritic Cell Algorithm for Mobile Phone Spam Filtering,” in *Proceedings of the 6th International Conference on Ambient Systems, Networks and Technologies Conformance (ANT)*, June 2015.
- Submitted journal paper entitled “Spam Filtering Framework for Multimodal Mobile Communication Based on Dendritic Cell Algorithm”