MapReduce based RDF Assisted Distributed SVM for High Throughput Spam Filtering

A Thesis submitted for the Degree of

Doctor of Philosophy

By

Godwin Caruana

Department of Electronic and Computer Engineering

School of Engineering and Design

Brunel University

May 2013

Abstract

Electronic mail has become cast and embedded in our everyday lives. Billions of legitimate emails are sent on a daily basis. The widely established underlying infrastructure, its widespread availability as well as its ease of use have all acted as catalysts to such pervasive proliferation. Unfortunately, the same can be alleged about unsolicited bulk email, or rather spam. Various methods, as well as enabling architectures are available to try to mitigate spam permeation. In this respect, this dissertation compliments existing survey work in this area by contributing an extensive literature review of traditional and emerging spam filtering approaches. Techniques, approaches and architectures employed for spam filtering are appraised, critically assessing respective strengths and weaknesses.

Velocity, volume and variety are key characteristics of the spam challenge. MapReduce (M/R) has become increasingly popular as an Internet scale, data intensive processing platform. In the context of machine learning based spam filter training, support vector machine (*SVM*) based techniques have been proven effective. SVM training is however a computationally intensive process. In this dissertation, a M/R based distributed SVM algorithm for scalable spam filter training, designated *MRSMO*, is presented. By distributing and processing subsets of the training data across multiple participating computing nodes, the distributed SVM reduces spam filter training time significantly. To mitigate the accuracy degradation introduced by the adopted approach, a Resource Description Framework (RDF) based feedback loop is evaluated. Experimental results demonstrate that this improves the accuracy levels of the distributed SVM beyond the original sequential counterpart.

Effectively exploiting large scale, 'Cloud' based, heterogeneous processing capabilities for M/R in what can be considered a non-deterministic environment requires the consideration of a number of perspectives. In this work, *gSched*, a Hadoop M/R based, heterogeneous aware task to node matching and allocation scheme is designed. Using *MRSMO* as a baseline, experimental evaluation indicates that *gSched* improves on the performance of the out-of-the box Hadoop counterpart in a typical Cloud based infrastructure.

The focal contribution to knowledge is a scalable, heterogeneous infrastructure and machine learning based spam filtering scheme, able to capitalize on collaborative accuracy improvements through RDF based, end user feedback.

Acknowledgements

I am grateful to all of my colleagues and friends at Brunel University who supported me in different ways during the PhD research. I am thankful to all the staff at the Department of Electronic and Computing Engineering, Brunel University.

I am heartily thankful to my supervisor, Dr Maozhen Li who supervised my research and guided me with great dedication throughout. His view of research has proved invaluable to my dissertation and supporting work. His personality, together with his high standards for research, makes him an ideal advisor. To me, he is also a role model.

I am also appreciative to the Malta Information Technology Agency for supporting my Ph.D. studies.

Lastly, I owe my deepest gratitude to my close friends, work colleagues – particularly those within the Technology and Systems Governance Group, family, and especially my wonderful wife, without whom I would not have made it through my PhD studies.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

ANN	Artificial Neural Network
CAPTCHA	Completely Automatic Public Turing test to tell Computer and Humans Apart
DAML	DARPA Agent Mark-up Language
DHT	Distributed Hash Table
DKIE	Domain Keys Identified Email
DNSBL	DNS-based Black-hole List
DT	Decision Trees
ECT	Estimated Completion Time
ESP	Email Service Providers
IAPP	Integrated authentication process platform
IDS	Intrusion detection systems
JVM	Java Virtual Machine
KKT	Karush-Kuhn-Tucker
kNN	K Nearest Neighbour
LCP	Lightweight currency protocol
ML	Machine learning
MPI	Message Passing Interface
MR	Map Reduce
MRSMO	Map Reduce Sequential Minimal Optimization
MTA	Mail Transfer Agent
MUA	Mail User Agent
OIL	Ontology Inference Layer
OWL	Web Ontology Language
P2P	Peer to Peer
RBL	Real-time blacklist
RDF	Resource Description Framework
S/MIME	Secure/Multipurpose Internet Mail Extensions
SASL	Simple Authentication and Security Layer
SIDF	Sender ID Framework
SMO	Sequential Minimal Optimization
SOA	Service Oriented Architecture
SVM	Support Vector Machine
TF	Term frequency
TF-IDF	Term frequency–inverse document frequency
URI	Uniform Resource Identifier
VSS	Vector Space Search
W3C	World Wide Web Consortium
XML	Extensible Mark-up Language

Authors Declaration

The work described in this thesis has not been previously submitted for a degree in this or any other university and unless otherwise referenced it is the author's own work.

Statement of Copyright

List of Publications

The following papers have been accepted for publication as a direct or indirect result of the research discussed in this thesis.

Journal Papers:

G. Caruana and M. Li, "A survey of emerging approaches to spam filtering," ACM Computing Surveys, vol. 44, no. 2, pp. 1–27, Feb. 2012.

G. Caruana, M. Li, and Y. Liu, "An Ontology Enhanced Parallel SVM for Scalable Spam Filter Training," Neurocomputing, no. 108, p.45–57, 2013.

Conference Papers:

G. Caruana, M. Li, and H. Qi, "SpamCloud: A MapReduce based anti-spam architecture," in 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery, 2010, vol. 6, pp. 3003–3006.

G. Caruana, M. Li, and M. Qi, "A MapReduce based parallel SVM for large scale spam filtering," in Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on, 2011, vol. 4, pp. 2659–2662.

CHAPTER 1 - Introduction

From just an annoying characteristic of the electronic mail epoch, spam has evolved into an expensive resource and time consuming problem. Spam continues to contaminate legitimate email communication and its enabling infrastructure. The costs of processing power, storage, as well as the human effort required to deal with spam are substantial.

Spam varies in shape and form [1]. Nonetheless, it tends to exhibit a number of similar traits in terms of structure, content, and diffusion approaches. There is a perceptible business reason and justification for its proliferation. From a spammer's perspective, the effort and cost of sending a substantial number of emails is minimal, yet the potential reach in terms of the magnitude of the available audience size is enormous [2]. The prospective profitability is clearly described by [3].

Unless careful mitigating controls and filtering considerations are applied from the outset, the value proposition of email and associated services will be reduced over time. This degradation is contextualized in terms of overall usability and value for money, as spam starts to deeply permeate respective email infrastructures. Consequently, approaches to spam filtering have been continuously researched, explored and applied with varying degrees of success. The techniques applied and employed by spammers continue to get smarter as well, with the primary intent being to outsmart and out-compute counterpart detection and filtering schemes.

To this extent, the overall objectives for this research are therefore to:

- Establish the current research landscape in terms of spam filtering techniques from the various perspectives involved, including algorithmic, architectural and trends.
- Design and implement a high throughout spam filter training approach able to capitalize on distributed computing techniques.
- Evaluate the distributed spam filter training scheme effectiveness in terms of performance and accuracy and compare it with relevant current research.

- Identify and evaluate how end user contribution can be considered and employed to improve spam filtering accuracy, using a feedback loop approach for the spam filter training scheme adopted.
- Research and explore the opportunity to exploit heterogeneous, commodity based, large scale computing environments to perform spam filter training in a high throughput construct.
- Evaluate the heterogeneous optimization approach(es) explored from a performance and cost perspectives.

1.1 Challenges

Velocity, variety and volume are key characteristics of the modern day spam challenge. The rate of spam proliferation (velocity), the continuously changing payload and approach type (*variety*) and sheer numbers (*volume*) have reached unprecedented levels. These attributes are indicative of the scale of the problem – one which can be characterised as an Internet scale problem. This mandates Internet scale techniques, algorithms and approaches for segregating ham from spam – and which can evolve at Internet speed.

Accuracy and performance are two fundamental properties for effective spam filtering. Establishing the right balance between them is non-trivial and not guaranteed completely accurate. This is a key challenge in any spam filtering construct. False-positives which can be very costly. On the other hand, email being highly entrenched within the general Internet ecosystem as it is, any spam filtering approach needs to make sure that it is as non-intrusive as possible in order to be of any use. This constrains the opportunity areas that can actually be exploited for improving spam mitigation – another fundamental spam filtering challenge.

From a wider perspective, spam filtering can be represented as a text classification challenge. Machine learning has been applied extensively in the context of spam filtering. Whilst numerous machine learning techniques are widespread, Support Vector Machines have and continue to garner increased attention in this construct. However, SVM approaches are inclined to be computationally expensive, especially when the size of a training data set is large. The scalability of typical SVM classifier training still remains an open challenge by virtue of the involvement of constrained, convex, quadratic programming challenges. In simpler terms, this refers to the degree of computing time required which is quadratic to the number of training data elements. This challenge is amplified further in a spam filtering construct due to its variety and velocity (*as in rate of change*) attributes.

Furthermore, this fluctuation of spam in terms of volume amplifies these issues considerably. Whilst stale techniques become obsolete quickly in such a fast moving landscape, without the ability to scale, up and down, spam filtering strategies can become very costly, quickly. This highlights yet another challenge - the evolution tempo spam of delivery approaches is alarming – with parasitic spam becoming increasingly evident. Other challenges include the widespread dependence on specialized complex algorithms and processing environments rather than simple, commodity processing resources based approaches. There is also a tendency to require an in depth understanding of how to tune and parameterize solutions. This both from an algorithmic as well as processing environment perspectives.

Such complex and tightly woven challenges continue to fuel spam filtering research from a number of standpoints – approaches in this context can be either based on specific areas or employed concurrently.

1.2 Motivation

Applied from a technical perspective (*other approaches, exist, including regulatory and organizational for example*), filtering approaches take various avenues. Common methods include email client extensions (*filters*) and filtering processes at the mail service-provider end. Simple Mail Transfer Protocol (*SMTP*) and machine learning based approaches are popular implementations of filtering schemes [4]. The application of heuristics, based around techniques such as black/white listing (*the application of trusted and non-trusted sources*), is also common [5]. Machine-learning techniques involve the analysis of email, mostly at content level, and employ algorithms, such as Bayesian, Support Vector Machines, and others, to segregate spam from legitimate email. These approaches have been extensively applied in spam filtering and exhibit different capabilities [6]. Schyryen [7] asserts that combinations of techniques and algorithms employed together, or cocktails, as more creatively referred to in [8], significantly increase the potential to alleviate spam-related issues.

As indicated earlier, Support Vector Machine (*SVM*) techniques have received increasing attention from the research community in the context of classification [9] [10] [11] [12] [13]. The qualities of SVM based classification in general have been proven remarkable [11] [14] [15] [16] [17] [18] [19]. SVM based approaches have also been proved to perform consistently well for spam classification [9] [20] [21]. There are numerous contributing factors to SVM's popularity for general machine learning problems as well as spam classification. Perhaps the more prevalent include the adaptability to different types of input data as well as limited sensitivity to feature selection [22]. The accuracy and performance of SVMs can also be further improved with numerous methods such as outlier removal, scaling and parameter optimization selection [23] [24] [20]. Improving the basic SVM approach by coming up with specific decompositions and solvers is another widespread technique [11] [25]. Another practice is to split the training data and use a number of SVMs to process the individual data chunks, subsequently aggregating results using various strategies [26]. Other approaches include the utilization of specialized processing hardware, such as Graphics Processing Units (GPU)'s [19] [16]. These are employed to offload speed sensitive algorithmic steps or used for the entire algorithm.

Distributed and parallel computing principles are believed to be key enablers for providing adaptable, Internet-scale anti-spam solutions. Whilst distributed and parallel programming in distributed environments is not a trivial task, various frameworks and models are available that make distributed and parallel computing more accessible. The MapReduce framework is one such programming model intended to abstract large scale computation challenges and enable a degree of automatic distribution and parallelization [27]. MapReduce was popularized by Google and primarily motivated by the need to be able to distribute and parallelize the processing of Internet scale data sets. Popular implementations include Mars [28], Phoenix [29], Hadoop [30] and Google's implementation [31]. Whilst the consideration for and application of MapReduce based approaches in the context of distributed systems is hardly new, it is also an area which one can safely assert as being relatively 'fresh' in terms of exploiting its potential for application in specialized ways. This includes uniting it with modern machine learning algorithms for spam detection and filtering, such as support vector machines [21]. To date, MapReduce's primary application is towards data intensive tasks rather than for computation in general.

Efforts to the latter's extent do however exist, including the work presented in [32] [33] [27] [34]. From this standpoint, MapReduce can be employed for dealing with both data and computationally intensive challenges. It is also believed to provide a motivating research prospect by virtue of the specific characteristics associated with this (*spam filtering*) problem. Tuning contemporary and innovative machine learning based spam filtering techniques to this metaphor is believed to provide ample potential for research.

In the application of a distributed spam filter training architecture and in the context of MapReduce, respective ‘Mappers’ and ‘Reducers’ will perform the steps required for machine learning computing sequences. It is expected that distributing spam filtering jobs and spam data to a potentially ‘unlimited’ number of computing resources can effectively tackle the spam challenge in a large scale construct. MapReduce can thus be employed as an enabling technology to facilitate high performance machine learning techniques for spam filter training/filtering. As already indicated, SVM training is well known to be a compute intensive task because of the quadratic programming challenge it involves, i.e. training time increases exponentially with the number of training elements involved. However, distributed MapReduce based SVM training would scale well because a number of nodes can be utilized concurrently and each node can deal with a subset of the training data at an individual level rather than processing the entire data set, thus alleviating this drawback. This is the basis of the distributed SVM, designated MRSMO, designed and evaluated in this dissertation.

However, the application of distributed computing techniques to what is normally a sequential algorithm, as is the case of many standard SVM formulations, can have side effects. The computation of SVM support vectors is a global optimization problem. One of the potential challenges of the decomposition and subsequent reconstruction of SVM’s to be considered in a distributed computing construct is degradation of classifier accuracy. This is where RDF based techniques are believed to provide an opportunity to mitigate this challenge. Ontologies enable the re-use of domain knowledge and expertise by formal and unambiguous concept representation. They provide a sound basis for formal information exchange and automated processing between machines, including varying degrees of automated ‘reasoning’. In this research work, an RDF based, intelligence supplementing and training instance correction feedback loop is explored as a potential means to mitigate the accuracy degradation exhibited by the reformulation of a sequential SVM in a distributed construct.

In Cloud computing environments, the ability to scale up and down as well as potential financial benefits, such as reduced capital expenditure costs, are touted to have created a market worth hundreds of billions [35]. Cloud computing environments such as Amazon Web Services [36], Windows Azure [37] and the Google Cloud Platform [38] are heavily based on highly heterogeneous, multi-core, distributed and parallel computing models and virtualization technologies. Heterogeneous environments are reflective of continuously evolving technology. Maximization of computing capability imbalance is considered very important in Cloud Computing environments. Despite continuous evolution [39][40], cloud services, cost and chargeback models can be considered still relatively immature [41]. Establishing a balance between performance and cost in such a context is considered a challenge [42]. The underlying mechanics of multiplexing and resource pooling - key cloud computing characteristics - can impact the actual runtime performance of virtual heterogeneous resources. This challenge is further amplified by the actual host behaviour, including operating system scheduling, workload, performance characteristics etc. In this work, a heterogeneous aware, MapReduce based task to node matching and allocation scheme is presented. Via configuration, the proposed approach is able to establish a degree of balance between performance and cost.

Summarizing, spam filtering research is an active study area. Whilst research on SVM based spam filter training has been carried out from various perspectives, a significant number focuses on specialized SVM formulations, solvers and/or architectures in isolation. This is referring to the emphasis on specific and specialized performance enhancement areas rather than overall flexibility, ease of use and practicality. There is also a tendency to rely on specific strategies and techniques that are mostly intended to target specific scenarios. Specialized configurations (*including special hardware use, special software stacks and/or specialized code bases*) limit applicability for practical large scale spam filter training and classification. The use of specialized hardware and environments does not allow common computing resources to be employed easily. When special architectures and/or configurations are employed, improving on performance via the use of additional, commodity computing resources/nodes is not always viable. It is also much more complex to scale well at the rate mandated by continuously changing spam traffic and patterns. Furthermore, special stacks and configurations require equally specialized knowledge for operation.

1.3 Main Contributions

The challenges, and subsequently opportunities, highlighted in section 1.1 form the basis of the research effort proposed in this work. The focus is to try to tackle a number of critical success factors with respect to practical, large scale spam filter training using commodity infrastructures in a high throughput distributed computing construct. The primary goals are scalability, accuracy, simplicity and cost effectiveness. This in terms of algorithmic approach as well as the use of easily accessible, commodity hardware and software elements which provide real-world scalability and large scale data processing capabilities. As indicated in the abstract, the underpinning motivation and principal contributions can be summarized as a scalable machine learning based, spam filter training architecture, designed to capitalize on heterogeneous infrastructures and collaborative accuracy improvement efforts through end users, and, in the future, machine to machine, feedback loop.

To this extent, this dissertation work contributes a number of novel elements to research in the area of high performance, scalable spam filter training and thus, classification.

This research performs and provides an extensive literature review, in the shape of a survey of literature on emerging spam filtering techniques. A comparison of spam filtering research work focus, categorized in a number of key areas, including algorithmic, architecture and trends is provided. The various research perspectives available are explored and discussed, covering traditional whilst focusing on emerging techniques and architectures, more specifically:

1. SMTP approaches, including DSMTP, ASMTP as well as grey listing and black listing studies are reviewed.
2. Explores research work available in the traditional heuristic and signature based areas as well as performs a study of the machine learning work in this context. Specific areas in this respect include the application of neural networks, Bayesian as

well as more recently Support Vector machines. Research in the area of accuracy, performance and scalability is also performed and discussed.

3. A study focusing on emerging trends, including peer to peer, grid and ontology semantics is contributed. Research work in the area of personalization, security and privacy is also discussed.
4. The survey work also presents an assessment of the key features of prevalent approaches in the form of techniques applied (*e.g. pre-Send / post-Send, machine learning, heuristic, signature based etc.*). The comparison also includes implementation complexity, performance, reach, quality characteristics and architectural footprints amongst others.

MapReduce is established as good candidate enabling framework for distributed SVM based Sequential Minimal Optimization (*SMO*) algorithm. This dissertation work decomposes a typical SMO algorithm and employs it as a baseline for the design of a distributed SVM, designated Map Reduce Sequential Minimal Optimization (*MRSMO*). This research also:

1. Implements the designed distributed SMO in a Hadoop M/R construct.
2. Subsequently evaluates the baseline sequential SMO with the presented distributed equivalent. The evaluation is carried out from a performance and accuracy perspective.
3. Employs two key sets of SVM kernels for the evaluation, namely Gaussian and Polynomial and demonstrates the significant improvement towards MRSMO across the tests performed.
4. Compares the performance of the distributed SMO with a baseline MPI work.

Whilst the distributed SMO (*MRSMO*) outperforms the corresponding sequential implementation as intended, the accuracy of the resultant classifier suffers. To this extent, this research:

1. Explores the possibility of using traditional ensemble and subsequently RDF based approaches for improving the overall accuracy.
2. Introduces an RDF based feedback loop with the M/R based distributed SMO.
3. Augments the training set with end user contributed '*knowledge*' by transforming the training data into an RDF graph representation. Misclassified elements are

identified using automatically generated SPARQL queries, optionally assisted with an intermediary C4.5 classifier where the respective decision tree fact rules are translated into SPARQL queries.

4. Employs Protégé for the proposed RDF based feedback loop improvement and assistance. Subsequently, the resultant, knowledge supplemented RDF base, is re-transformed to respective SVM (distributed MRSMO) training instances.
5. Evaluates the RDF assisted feedback loop accuracy improvement using both the SpamBase as well the TREC datasets. The improvement level reaches, and surpasses, the original sequential SVM accuracy.

In order to further improve the overall value proposition for a '*turnkey'* scalable anti-spam filtering architecture, this research also proposes an improved task matching and node allocation for MapReduce, using the out of the box Hadoop scheduler as a comparison baseline. In this respect, this research work contributes:

1. The design of gSched – a heterogeneous aware M/R task to node matching and allocation scheme for Hadoop. The focus of the design's parameters is on balancing cost and performance. A specifically configured Amazon AMI is also configured and discussed.
2. A comparison and evaluation of gSched on a virtual cluster using the baseline DFSIO and MRBERNCH Hadoop M/R benchmarks.
3. Comparison and evaluation of gSched on Amazon's EC2 cloud infrastructure, again using the TestDFSIO and MRBERNCH Hadoop M/R benchmarks.
4. Evaluation of the distributed M/R SVM (*MRSMO*) using the Gaussian and Polynomial Kernels - comparing the standard Hadoop and gSched task allocation scheme performance. These test are performed again on Amazon EC2.
5. Evaluation of gSched's performance, cost and scalability improvements using the standard WordCount test on Amazon EC2 nodes.

In summary, the principal contribution of this dissertation is a scalable, heterogeneous aware, machine learning based spam filter training architectural pattern, able to capitalize on collaborative feedback. This is corroborated by an in depth survey of emerging spam filtering approaches, the design and implementation of a distributed support vector machine algorithm and an RDF semantics based, end user assisted feedback loop for accuracy improvement. It is also supported by a heterogeneous aware, task to node matching and allocation scheme for MapReduce based spam filter training.

1.4 Structure

This dissertation is organised as follows. The initial research background, alongside the motivation are presented in this chapter. The primary contribution of this research is also articulated in Chapter 1.

Chapter 2 provides an extensive survey of current literature related to spam filtering. It provides an in depth discussion of the current as well as emerging concepts and application of spam filtering approaches, techniques and architectures. It also discusses the challenges and opportunities presented with respective combinations, focusing on emerging approaches. Chapter 2 also establishes the baseline principles for the consideration of specific underlying architectures for high performance spam filtering.

Subsequently, in Chapter 3, a brief discussion of key machine learning approaches is presented, focusing on the support vector machine perspective. MapReduce - an architecture for internet scale compute and data intensive processing is also presented. In this chapter a typical sequential SMO algorithm is decomposed and re-formulated in a distributed construct. This leads to the design and proposal of MRSMO. A number of experiments are performed and presented in this construct. The accuracy and efficiency of the sequential SMO are presented. This is used as a baseline for comparison with the corresponding distributed M/R implementation proposed. MRSMO is also evaluated with a baseline MPI implementation using two flavours of the SVM kernel, namely Gaussian and Polynomial. A number of simulation comparisons to establish the scalability of the proposed approach are also performed.

In order to further improve the accuracy of the distributed SVM, the application of RDF based techniques to be used for end user feedback contribution is evaluated. This is the basis of Chapter 4. In this section, SPONTO - Spam Ontology, based on the primary data set used for research in this context, namely SpamBase, is presented. The ontology reflects the basic elements in this specific data set, but also includes additional attribute assertions. The end user is able to contribute knowledge via the RDF based feedback loop. The instance data is used to generate baseline RDF graphs which are subsequently employed to establish misclassified elements. The original accuracy outcome based on the distributed SMO with the RDF supplemented approach is compared. The result of this work in this respect establishes accuracy improvements of 5%, on average, beyond the original distributed SMO's performance in this regard. An experimental test on the TREC dataset is also performed. The accuracy improvement, or rather degradation induced due the high throughput, distributed SMO, improves by an average 3%.

In Chapter 5 the distributed, RDF assisted spam filter training approach presented is taken a step further. A limitation of the default Hadoop MapReduce scheduling implementation employed is established. The out-of-the-box task allocation scheme assumes a homogenous underlying computing resource environment. The design and implementation of a heterogeneous aware Hadoop MapReduce task to node matching and allocation scheme is thus presented. Designated gSched, this approach is intended to establish a balance between performance and cost effectiveness in what has become a very popular computing resource provisioning construct, namely cloud computing. The task to node matching and allocation scheme is base lined and performance compared with the standard scheduler performance in an Amazon EC2 construct. The outcome of the evaluation of the two schemes in terms of performance when using the distributed SVM, Gaussian and Polynomial kernels are presented. The standard WordCount test to establish the scalability improvements between the original and proposed scheme is also performed and presented in this chapter.

This dissertation concludes with Chapter 6, summarizing the contributions, limitations and boundaries of this research as well as providing a number of suggestions for future improvements and extensions to this work.

CHAPTER 2 – Spam Filtering Approaches

There have been numerous efforts to review spam filtering methods [43] [4], and with varied scope. The increased velocity, volume and variety of spam demands high performance filtering algorithms and methods. The past years have witnessed a number of emerging technologies in computing, notably, peer-to-peer computing [44], [45], semantic web [46], and social networks [47]. These technologies facilitate the development of collaborative computing environments for solving large scale problems. A number of approaches, building on these emerging computing technology trends towards spam filtering have been more lately proposed and implemented.

This chapter provides an extensive review of spam filtering directed literature and is organized as follows. Section 2.1 provides an overview of traditional approaches to spam filtering. Section 2.3 reviews emerging spam filtering approaches and discusses the challenges that these approaches present. Section 2.4 discusses various issues related to security and privacy in spam filtering. Section 2.5 presents a number of observations identified in this survey and discusses research directions, closing off with a summary.

2.1 A Synopsis of Traditional Approaches to Spam Filtering

As briefly cited in the background, SMTP and machine learning are popular implementations of spam filtering approaches. An overview of these is provided, accompanied by a discussion of a number of core challenges associated with them in the next sections.

2.1.1 SMTP Approaches

The SMTP protocol is at the foundation of the email enabling infrastructure. It is regularly argued that a number of the limitations capitalized on by spammers are inherent to its original design [48], since one of the primary original concerns was to preserve simplicity in exchange. The motivator behind the work presented in [49], for example, is primarily focused on addressing a number of these simplicity limitations. The basic approach presented tries to shift the control from the sender to the receiver in order to mitigate the unwarranted reception of unsolicited mail. This is achieved by changing, in part, the delivery approach from sender-push to receiver-pull [49]. The proposed extension to the baseline protocol, designated Differentiated Mail Transfer Protocol (*DMTP*), introduces an additional measure for email transactions. The recipient is first sent an '*intent'* message [49], subsequent to which the message content is only retrieved if interest in the former is shown. While this approach exhibits substantial potential, any change to the underlying core email transmission infrastructure is bound to introduce operational complexities. Furthermore, it is potentially prone to impact a number of intertwined enabling services, even if an incremental deployment approach is undertaken, as suggested. This risk has the potential to considerably impact and limit adoption and acceptance levels for such an approach.

The SMTP protocol's core inherent simplicity is also evident from the very small set of operations allowed, namely HELO, MAIL FROM, RCPT TO, DATA, . , QUIT, and RSET, in that order. To this extent, Li and Mu [50] base their spam-filtering approach by scrutinizing SMTP traffic payload data. More specifically, abnormal SMTP interaction '*patterns'* are considered. It is argued that high throughput access and interaction patterns are not representative of normal email users or email clients' communication exchanges. This motivation is subsequently employed as the basis for assuming that such traffic is most likely spam. Designated Abnormal SMTP Command Identification (*ASCI*), the results from the work presented in [50] indicate that this approach can achieve around 11% reduction in spam. The endorsement of normal protocol interaction by spammers or bots will undoubtedly hinder the spam-sending throughput rate considerably. However, adopted as is, ASCI runs the risk of not retaining original effectiveness over time. This challenge manifests itself when intelligent botnets are involved and able to identify and react to basic non-delivery issues for example. Irrespectively, this approach provides a representative first level of defence in reducing the number of spam that actually infiltrates inwards from the edge network.

Another prevalent technique for filtering spam is Greylisting. Whilst numerous variations exist, the core concept is rather simple. The IP address of the SMTP host participating in the exchange and the addresses of both the sender and recipients (*referred to as a triplet*) are employed to deny the email exchange from happening, initially. Triplet information is stored and subsequent attempts with the same signature are allowed to perform the mail exchange. This approach is based on the rationale that most spammers do not care to retry sending the same email from the same relay. Numerous studies supported by empirical results [5] [51] [52] show that this approach is rather effective. Furthermore, it is non-intrusive from an end user perspective, which is considered a critical success factor. The combination of Greylisting with other techniques, such as Blacklisting, further increases the potential to control spam. Blacklisting, or real-time blacklist (*RBL*), involves setting up and maintaining a list or lists of sources, including open relays that are identified as spam propagation sources. Numerous sources for Blacklists and associated services exist, notably Spamhaus [53], SpamCop [54], and DNSBL [55]. Innovative variations of Blacklisting-based approaches exist as well [56].

Non-SMTP approaches, which can be classified as infrastructure related, but are not necessarily directly related to SMTP, also exist. Spam can be stopped at the source before it leaves its origin, in transit, or at the receiving end - various degrees of filtering and combination points exist in this construct [57]. Approaches include those which sit beyond the individual point of spam origin, but still within the source network. Other inspection and filtering points sit at the very edge of the destination email service-provider network. Gateway and router based approaches are popular as well. One such example is presented and discussed in [58] . One can refer to this approach as late pre-send given that filtering happens just before the email is dispatched out of the originating source network. Intrusion detection systems (*IDS*) are regularly employed near the entry point of enterprise networks. The method presented in [58] capitalizes on this approach, albeit this method could also be employed at different parts of the network. By inspecting various information sources that the IDS gathers, assembles, and has access to, as well as the ability to identify correlations, the approach performs dynamic, domain specific blacklisting. IP address domains which are identified as spam sources are blocked via the automated insertion of respective firewall blocking rules, based on the intelligence gathered and built by the environment. This approach avoids spam flowing into the network in the first place. However, the effectiveness of this approach depends on a number of sources which are not fully under the control of the solution itself. This may severely hinder its overall effectiveness in a high throughput production system.

2.1.2 Machine-Learning Techniques

A number of filtering algorithms are commonly employed from a technology standpoint, varying in complexity and effectiveness. However, spam filtering algorithms can be split into two overall umbrella approaches, namely machine and non-machine learning methods. Approaches applied in the former category include Bayesian classification, neural-networks, Markov-based models, and pattern discovery. Rule, signature and hash-based identification, blacklisting, and traffic analysis, among others, are typical techniques that are employed with respect to non-machine learning variants. Both classes have their advantages and disadvantages and demand different requirements in terms of processing power and bandwidth for example.

Machine learning variants can normally achieve effectiveness with less manual intervention and are more adaptive to continued changes in spam patterns. Furthermore, they do not depend on any predefined rule sets analogous with non-machine learning counterparts. Two principal machine learning approaches for inferring spam classification in a semi or fully autonomous fashion are commonly considered, namely supervised and unsupervised. The former depends on an initial training set to assert classification, while the latter does not, employing rather other techniques, such as clustering, to achieve its objectives. In supervised learning, model input observations, more commonly referred to as labels, are associated with corresponding outputs upfront; in non-supervised approaches, any observations are associated with latent or inferred variables.

From a high level perspective, considering spam from a text classification dimension in the context of machine learning, one can describe the basic problem via the dichotomy $m_i \in \{-1, +1\}$. Here m_i represents a set of messages and −1 and +1 represent ham (*non-spam*) and spam respectively. In order to apply text classification to spam, messages are normally represented as a set of vectors, such as $\{\} : m \rightarrow R_n$. Here the vector set R represents the features of a message. Each message has a corresponding feature vector constructed using respective features. Identifying those features of a message which have the qualities to indicate whether it is spam is a critical task for machine learning approaches. This also includes, where applicable, a number of pre-processing tasks, such as lexical analysis and dimensionality reduction, in the form of stemming, cleansing, and normalization [59]. Features can take the form of words, combinations of words and phrases, etc. Generally speaking however, fewer features normally represent greater generalization and better performance. However, this mostly at the expense of not being able to obtain the required class separation, that is the identification of the optimal separating level between the classes (*spam/ham*), thus minimizing or removing entropy. Various schemes intended to identify the best features in terms of quality and number are possible, as discussed by [60]. Feature vectors are also typically associated with weights intended to influence the outcome of the classification accordingly. Popular weighting schemes include term frequency (*TF*), binary representation, and term frequency–inverse document frequency (*TF-IDF*) [61].

Besides the definition of the respective feature vectors and subsequent creation of the training set, there is also the classification algorithm itself that evidently needs to be considered. As indicated earlier, numerous machine-learning algorithms exist, including Decision Trees (*DT*), Bayesian classifiers, Artificial Neural Networks (ANN), and Support Vector Machines (SVM). While the specific review and discussion of machine learning based approaches and their accompanying algorithms is not the scope of this work, these approaches continue to garner a lot of attention in the context of spam filtering [43] [4] [62] [62].

2.2 General Challenges

Although significant advancements in spam filtering have been made with traditional approaches, a number of challenges remain [57]. SMTP is at the core of the email enabling infrastructure, so it follows that considerable effort to handle spam at this level has been applied, including research from various perspectives [49] [63]. The biggest challenge for SMTP based approaches is to ensure that they do not impact in any way the underlying enabling infrastructure. The number of components and building blocks involved in successful email exchange and which are directly dependent on the protocol itself are not trivial. While the potential is there, technical complexities as well as financial challenges restrict (*in various ways and to different extents*) the widespread consideration and adoption of SMTP-based approaches. As an example, email exchange-path verification, which provides the ability to trace the real origin of an email sender, requires appropriate accounting in the context of the packet network involved. Authentication, on the other hand, requires appropriate client support as well as cooperation, and therefore may limit the clients that are able to interact in such fashion. While the popularity of Blacklisting and Whitelisting continues [4] [57] [64] [65] challenges still exist, including the increased possibility of blocking legitimate email exchange, or false positives, which is considered to be very costly. Manual interventions for applying changes or adding new records to these lists make them prone to mistakes. There is also an additional burden of keeping them up-to-date. Similarly, where heuristic based approaches are employed, it is difficult to maintain the necessary patterns that are employed for matching up-to-date.

Different challenges also exist in machine learning approaches to spam filtering [57]. Specific algorithms have particular advantages and disadvantages which influence their overall accuracy and performance. This has been extensively discussed in numerous related works which compare algorithms and approaches from a research [22] [66] [59], as well as real-world application perspectives [67] [68]. From an end user standpoint, what the email user perceives as spam and not spam also remains an active research question, by virtue of the degree of personal subjectivity associated with its classification. Challenges surrounding false positives, including the implications they can lead to, have also been discussed extensively. False positives refer to wrong outcomes from classification schemes. More specifically, they portray the situation whereby ham is classified as spam. While such challenges can be somewhat mitigated by the concurrent application of a number of different classification schemes (*classifier combining*) as discussed in [69], they still constitute an on-going challenge. The same applies to security and other typical challenges associated with machine learning [70].

Furthermore, in a machine learning context, the overall utility of a classifier directly depends on the training set [71]. The training element of many machine learning approaches requires content from real world email in order to be as representative as possible. Quality training data may not be readily available or tailored to an extent that may impact the overall quality of the learning models. This challenge is amplified further when the learning data is frequently changing and mandates continued efforts retraining it. The performance of numerous machine learning approaches is also largely dependent on the size of the training set. Although a larger, representative, training set would under normal circumstances produce better results, more processing time in terms of the learning process is normally required. Furthermore, the identification and selection of the best feature set introduces further challenges. Learning data tends to reflect specific attributes, perhaps not distinguishing enough between the relative weights of attributes (*and training instances*), as well as missing relevant properties. Changes in spammer behaviour, as well as the delivery type of spam payload, have also introduced different sets of challenges which require innovative approaches to remain effective in spam filtering.

Additionally, the traditional Mail User Agent and Mail Transfer Agent model for spam filtering faces continued challenges related to scalability and performance. The sheer number, type, and size of spam traversing communication exchange paths mandates considerable computing resource requirements for filtering. These requirements tend to be of an order of magnitude beyond most traditional filtering architectures. It is extremely difficult for traditional spam filtering architectures to be in position to scale up (*and down*) effectively and at the rate mandated by the fluctuations of spam proliferation.

Summing up, traditional approaches to spam filtering face continued and increasing challenges. These approaches are usually deployed on computer nodes which process spam individually without collaboration, limiting their application to a relatively small scale.

2.3 Emerging Approaches

Spam and spammer techniques evolve through time, capitalizing on new approaches and exploiting new flaws. Building on evolving developments of computing technologies - including peer-to-peer (*P2P*) computing, grid computing, semantic web, and social networks - a number of emerging approaches have been proposed for spam filtering. These approaches have the potential to tackle a number of challenges discussed earlier. They are also intended to improve overall spam filtering effectiveness. The following sections discuss representative research work in these areas.

2.3.1 Peer to Peer Computing

The ability to capitalize on distributed resources, including hardware, software, as well as human participation is what constitutes and drives the core of peer to peer (*P2P*) initiatives and architectures (*see Figure 2-1*). From a very high level perspective, there are two umbrella types of P2P architectures, namely centralized and decentralized. Decentralized P2P networks can be further classified into unstructured and structured P2P networks.

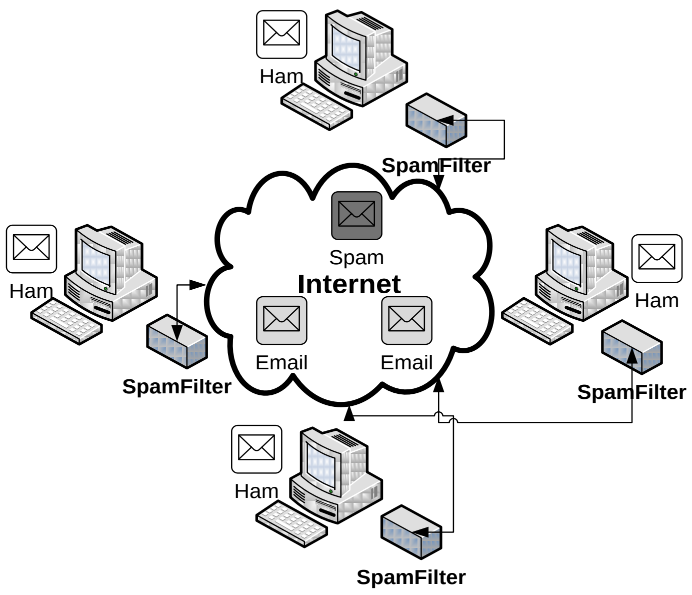


Figure 2-1: Peer-to-peer computing for spam filtering

P2P architectures form the backbone of numerous distributed, high-performance services and systems, significantly popularized in recent years by Internet-based application sharing communities and software. These include those based on the Gnutella [72] and FastTrack protocol (*undocumented formally*), such as Limewire [73]. The ability to exploit scalability easily, as well as offer a good degree of personalization, are inherent potentials of P2P-based systems. Furthermore, the ability to make efficient use of common interests is very inductive to collaboration. In this context, P2P computing application for spam filtering is applied in a number of ways. Peers can collaborate towards the identification and filtering of spam via the exchange of either computing power or, more simply, intelligence gathered on email and spam. Various techniques can be employed; however the algorithms and techniques employed in this context are normally focused on minimizing network bandwidth and identification of nearest peers. Exchange of information is normally focused on pre-computed signatures which relate to previously scrutinized and tagged content. The structuring element plays a critical role in P2P systems. Flat, hierarchical, and distributed hash tables (*DHTs*) are commonly considered. Specifically with respect to DHT structured P2P networks, popular implementations such as Pastry [74] and Chord [75] provide sound architectures that are scalable and are inherently fault-tolerant. [76] Provides a review on the application of P2P systems to spam filtering in this context.

The CASSANDRA (*Collaborative Anti-Spam System Allowing Node-Decentralized Research Algorithms*) initiative, for example, presents a personalized approach for tagging and classifying spam [77]. The work employs a P2P architecture and exploits computing resources that participating nodes can offer to achieve high scalability and flexibility. The authors argue that, whilst common spam filtering established on traditional non-machine / machine-learning approaches and based on mail content are reasonably effective, they are still considerably prone to false positives. Collaborative spam filtering tends to be better suited to tackling issues related to spam drift, or rather, spam and spammers tendency metamorphoses in trying to circumnavigate filters based solely on content. The prototype architecture described, however, does not seem to consider possible issues related to how it can distinguish between real-world users and automated spam bots.

Building on a Chord P2P network, [76] present another approach to exchanging partial and hashed message data on spam between participating nodes. This allows the network to distinguish good participation from malevolent. However, one challenging issue is that spammers could easily influence the P2P network by changing their participating status from non-trusted to trusted participating entities. In the real world, this may cause the approach to gradually loose efficiency over time. Being a proxy based implementation allows for great flexibility in terms of the usability from the widest variety of email clients (*assuming adequate levels of heterogeneity*). However, it also introduces an additional element of complexity, as well as computational resource requirements at the client end. This challenge may be further amplified by virtue of the proxy itself being a simple Web server, increasing the local node's attack surface from a botnet perspective. Updates to the proxy itself can also prove difficult given the potentially high distribution of nodes which will be making use of the localized proxy.

Other P2P based approaches to spam filtering include those presented by [78], [79], [80], and [81]. In [82], the authors discuss a diverse approach. The primary motivation of their work concerns the collaboration of email servers, rather than the collaboration between mail clients. Spam checking is primarily performed at two specific points in time: before being sent by the sending email service and at the receiving email service end. A system of rewards and restrictions is applied to participating servers. The overall standing of these servers with regards to trust directly effects the degree of influence on the resolution. A critical factor for the effectiveness of this approach is the participation of a considerable number of email service providers (*ESPs*). Effectiveness is drastically decreased if participation is low. Additionally, the possibility for legitimate email to be slowed down by virtue of the respective ESP being classified as a temporary spam source could potentially irritate legitimate email users. The work does, however, indicate possible avenues for further exploration. On the other hand, the work as presented does not consider the utilization of resources available at the client's end for additional collaboration and contribution towards the improvement of spam detection and filtering. It can therefore be argued that this reduces possible input sources, as well as overlooks the possibility of benefitting from additional processing power or intelligence for improving filtering quality.

A rather distinctive viewpoint using Percolation Networks as a theoretical base is provided by Kong et al. [83]. This approach identifies spam using a typical classification scheme (*such as Bayes*). A digest function, which the client must be able to generate, is subsequently produced if the email is identified as spam. The client then publishes this newly identified intelligence to a number of participating neighbours. This is done using a random walk path through the percolation network based on the graph edges (*distance*). Mail clients can query the network to identify whether an email has already been tagged as spam. This scheme works by implanting the inquiry using the same approach employed for publishing its spam digest database updates, that is, by traversing the percolation graph using a random walk to a pre-specified distance. The query percolates and aggregates digest hits across nodes, returning the results of the query via a reverse path. If the hit count meets a specified score, the message on which the query digest was based is declared as spam. Data exchange during querying and publishing does not appear to be high, but a degradation of performance from an underlying network perspective still has the potential to inflict non-trivial performance penalties, reducing the overall effectiveness of this approach.

Overall, in the context of spam filtering, P2P approaches provide a number of advantages by exploiting the typical underlying mesh based network infrastructure. P2P networks are well suited for collaborative spam filtering. The computing capacities of P2P networks can easily grow with the increasing number of peer nodes. Ad-hoc distribution of messages also eliminates single point of failures and permits active distributed processing. Another opportunity which can be exploited is aggregation of participants with similar interests in spam filtering. However, P2P networks suffer from a number of issues: unstructured P2P networks are technically challenged in achieving high scalability in locating peers. They are also prone to a number of security challenges. In index poisoning, for example, when a resource search is performed, the results returned may not reflect correct information because bogus information may have been inserted in the respective tracking structures. Also, structured P2P networks, while DHT approaches provide the necessary scalability, suffer from complex and challenging overheads associated with structure maintenance, as well as management [84].

2.3.2 Grid Computing

Originally conceptualized in 1997, grid computing has evolved into an effective computing paradigm for solving data and computationally intensive problems by utilizing various computing resources over the Internet [45]. Figure 2-2 portrays a general grid computing architecture for spam filtering. Grid computing promotes the utilization of collective computing resources that facilitate spam filtering on a large scale. Rather than investing in resources to keep up with continued spam increase, capitalizing on grid resources on a need basis has significant advantages. Hosted and cloud based anti-spam service implementations such as [85] and [86], in which the spam filtering element is contracted out to a third party, are also commonly based on grid computing principles. Similar to P2P computing, the ability to share resources, as well as intelligence in a collaborative fashion, provides further opportunities improving the quality of spam filtering.

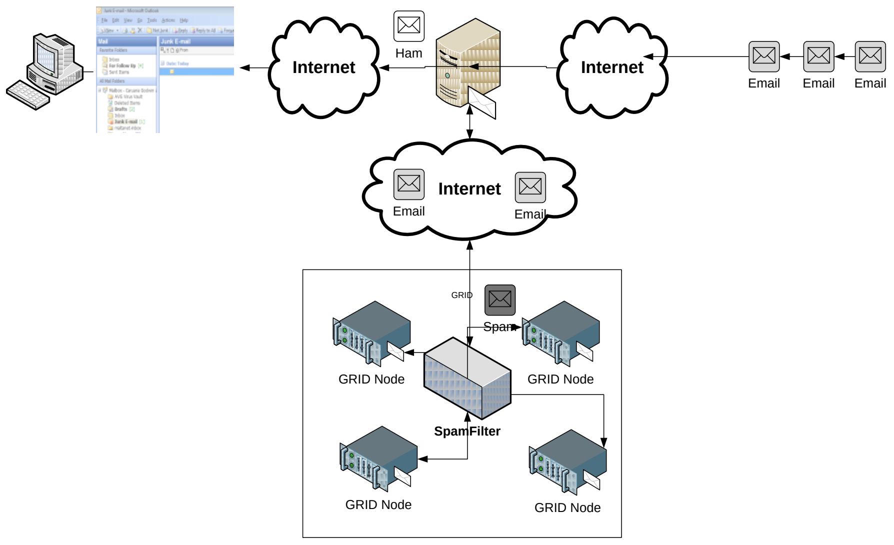


Figure 2-2: Grid computing for spam filtering.

The work discussed by [87] present a grid based distributed Bayesian classification scheme for spam identification in a collaborative fashion. The authors argue that the grid approach is more efficient than P2P-based counterparts. The key challenge with the proposed approach is the dependency on specific plugins for specific email clients, which is, arguably, architecturally brittle (*due to client stack and version differences*), as well as tightly coupled, which limits flexibility. The notion of a virtual organization in the context of grid architectures, which creates the opportunity for a worldwide anti-spam organization, is mentioned in this discussion. In such a scenario, the virtual organization can offer a number of associated services ranging from providing computing resources for spam filtering to legal frameworks for spam control. However, limited discussion regarding this is provided. Another similar grid based architecture initiative is discussed at a very high level in [88] which utilizes a global grid infrastructure for gathering spam intelligence.

In principle, grid based approaches benefit from theoretically 'unlimited' scalability potential. The same applies to the ability to execute jobs in a distributed computing construct in the context of machine learning for large scale spam filtering. The ability to use computing resources on a need basis is also considered an important attribute, by virtue of limited capacity planning visibility opportunities with respect to spam increase, decrease, and fluctuations. Furthermore, underutilized resources can be capitalized upon for high volume spam filtering, thus making effective use of 'idle' resources. However, a grid is a large scale computing environment in nature, and grid resources are highly heterogeneous, thus carrying an obvious element of complexity.

2.3.3 Social Networks

Social networks are arguably one of the most remarkable Internet phenomena of recent years. They have emerged as one of the most sought applications of the World Wide Web - a '*killer application*'. They provide a ubiquitous ecosystem which allows users to identify themselves, interact, share, and collaborate. Alongside similar work by [89] [90], in [91] the authors pose a research question in the context of spam using social networks for gathering intelligence. Analogous in a number of ways to PageRank [92], MailRank is introduced in this study as a ranking and classification scheme. Two approaches are described: one basic, the other offering personalization. In both cases, the primary technique is the aggregation of email intelligence from participants of the social network. A network graph is constructed relating scores to each respective email address. By employing a power iteration algorithm, a reputation weighting measure influenced by participating members via the exchange of trust votes is associated with emails is employed. This architecture relies on the introduction of the centralized service designated MailRank, as well as an intermediary proxy intended to sit between the Mail User Agent and the production email service.

Another application of social networks in the context of spam filtering is discussed in [93]. The primary motivator here is the exchange of filters between respective social network participants, rather than exchanging signatures or similar statistics to increase overall spam intelligence. The authors argue that there are considerable benefits from capitalizing on end user processing capabilities in contrast to purely centralized approaches. This work also introduces the notion of a spam filter description language and a pluggable engine able to make use of filters which adhere to the described specification. This creates a way to describe spam filters and associated behaviour in a uniform way, which can be subsequently exploited by architectures which are aware of its specification.

In [94], the authors evaluate Web 2.0 based anti-spam methods. Their result highlights the need for more sophisticated measures for combating spam in the social networking domain, whilst ensuring that any applied measure remains as non-intrusive as possible. The work presented in [95] also discusses an innovative approach for tackling spam in a social networking context. The primary focus and motivation here is the ability to categorize participants according to their characters using the concept of feature bundles. The conceptual features describe prototypes, which include events, actions, emoticons, spatial relationships, and social relationships. The final objective is to categorize users via two primary dimensions, namely sociability and influence. The former is employed as a metric describing the level and nature of information; the latter is based on the influencing element that the profile tries to carry. The approach employs combinations of these metrics in turn to identify the user type. For example, a user with low sociability and high promotion is initially evaluated as an entity whose primary intent within the network is marketing and expansion.

An approach for specialized social networks dealing with video content is discussed in [96]. The challenge addressed in this work is identifying whether video content is ham or spam.

The authors employ heuristics and a machine learning approach based on video attributes in conjunction with a user's behaviour to perform classification. Characterization is based on comments submitted and popularity in terms of number of views. Video spam is denoted if, for example, comments submitted are not related to the video description or have zero-length content. Here, the challenge of what is considered spam to whom emerges; that is, the degree of subjectivity of what is spam is even greater than textual counterparts because of increased difficulty and complexity in characterizing video content.

Overall, similar to P2P, social network based approaches yield a number of potential intelligence collection opportunities. Social networks also tend to aggregate participants with similar interests, promoting collaborative environments which benefit spam filter training. However, controlling and filtering spam in a social network context is generally more difficult compared to traditional email exchange ecosystems [97]. Given the complexity in identifying and segregating participants' intents, it is also prone to generic poisoning attacks [98]. Similar to P2P, these poisoning attacks refer to the intentional introduction of influencing factors intended to bias outcomes towards specific objectives which relate to the environment.

2.3.4 Ontology Based Semantics

Ontologies have played a critical role in the arena of the semantic web. They provide a formal basis for the definition of knowledge representation, subsequently enabling the exchange of this knowledge relatively easily. Ontologies are used to describe specialized domains and an associated set of vocabularies. Semantics relate to the ability to portray and understand the meaning of information in an expressive way. Their use in the context of spam filtering facilitates the definition and understanding of spam in a better and formal way.

On this rationale, the work articulated in Youn and McLeod [99] argues that the application of ontologies to formalize spam offers a sound basis and numerous opportunities for improving the definition and filtering of spam. This in the context of reflecting user preferences more appropriately for this particular work. Putting greater emphasis on the personalization aspect will increase perceived usability. Ontological knowledge can be built by identifying and formalizing the relationship between a user choices and how spam is reacted to, as presented for example in the work by Kim et al. [100]. The authors classify reaction into four types: Reply, Delete, Store, and Spam. A user profile is created by scrutinizing the personal actions in terms of email replies. On this basis, association mining is applied to an initial reference dataset using Weka [101]. The authors employ Karnaugh maps for logic simplification and subsequent axiom creation. The ontology inference engine developed and based on a first order logic reasoner classifies respective emails established on the profiled user typical interactions. This approach is somewhat similar to the work proposed by Balakumar and Vaidehi in [102]. Although the quality of the outcome (*in terms of correlating user preference with typical reactions to specific emails*) was constrained by the limited initial scope and entropy of the data set, the outcome still indicates the potential of ontologies for spam filtering.

'Grey' mail is a class of email that does not exhibit sufficient traits for establishing a degree of confidence that respective content is spam or ham from the outset. This challenge is further amplified from a personalization perspective, or rather how specific users identify specific email, given the associated subjectivity. These set of attributes make the study of ontology based approaches a motivating consideration in the context of grey mail. To specifically target Grey mail classification, Youn and McLeod present a spam filter based on a personalized ontology in [103], [104]. The ontology is built on specific attributes related to individual user preference such as user background, hobbies and other attributes [103], [104].

Hsia and Chen describe an approach for image based spam detection [105]. In this work, a scheme based on exploiting hidden topics within images is employed. These *'latent'* topics are identified (*and subsequently employed*) as training input for a binary classifier. The authors describe a probabilistic approach for inferring hidden semantic meanings represented as images. In Youn and McLeod [99] on the other hand, the image element of spam representation is tackled using a traditional approach based on optical character recognition, and term frequency - inverse document frequency (*TF-IDF*) feature set selection. Additional processing is performed to convert the model generated via the adopted machine learning scheme using Weka [101], to an Resource Description Framework (*RDF*) transformation. This step is employed in order to generate the required ontologies that are subsequently employed to create custom user filters.

The annotation of spam and email messages with metadata has additional benefits, including supplemented intelligence, context richness, and formalization. The incorporation of domain knowledge facilitates filtering processes, including training and classification for high accuracy in spam filtering. Furthermore, ontologies bridge the gap between the levels of understanding required for preparing training and classification models and the end user. By virtue of their inherent readability and expressiveness, ontologies also provide end users with the opportunity to understand and contribute towards improving spam filtering. To date, however, there is no single '*standard'* ontology for spam annotation. Furthermore, separate initiatives tend to develop distinctive ontologies, thus creating challenging situations with respect to ontological interoperability requirements. Ontology based interoperability is by no means trivial [106] –complexity becomes further amplified in the spam context given the subjectivity aspects.

2.3.5 Other Approaches

Spammer behaviour study is a very important source for gaining wider insight from an overall spam detection and filtering perspective. This may be supported by providing ecosystems that allow spammer roaming while ensuring appropriate levels of monitoring. Gathering as much information as possible on how spammers operate is considered fundamental to identifying mitigating factors [107] [108] [109]. Spammers tend to rely on aggregated intelligence to work out target email addresses. "*A taste of ones medicine*" approach can be used as an idiomatic narrative of the work presented in [110]. Albeit perhaps questionable from an ethical perspective, the work presented considers an original perspective, in an attempt to revert some of the advantages of a spammer's business model. The ultimate objective is to make spam more expensive or comparably less effective overall than alternate marketing schemes for example. The authors advocate that this will make spammers abandon their preferred activity of making money. Basically, one of the most important assets spammers have access to is a list (*referred to as a database in the text*) of email addresses, which defines the ultimate spamming audience. The work discusses the proposition of poisoning such a list in a collaborative fashion. The authors assert that the validity of these email addresses is close to 100%, and that a substantial reduction in quality would decrease the overall profitability of the spammer's business model. The suggested modus operandi is rather simple—inject routable fake email address to the database to inflate it with bogus records. They also suggest increasing respective anti-spam resources so that these newly acquired resources may be employed simply to mimic spam interaction, while, in fact, spam content is simply discarded rather than processed. This should result in less spam being received and processed by real end-users, assuming no change in the amount of spam being sent.

Shinjo et al. present an approach in which the filtering elements are considered as features that can be enabled and disabled according to a set of attributes [111]. These are referred to as '*capabilities'* exhibited by the email in transit. Capabilities are generated by a recipient and submitted to authorized senders from the context of the capability creator. If the email exhibits valid capabilities, the spam filter simply disengages its action and allows the email to be passed directly to the recipient's mailbox. One of the primary motivators behind this scheme is to eliminate false positive challenges. Because the implementation of capabilities is based on a standard approach using special SMTP headers, Mail Transfer Agent (MTA) and Mail User Agents (MUA) are able to handle or ignore the special directive(s) out of the box; however, extensions to the MUA are required to process the capabilities.

Esquivel et al. present another approach to limit the proliferation of spam from its sources [112]. This is based on passive TCP fingerprinting at the router level. Here, SMTP servers collaborate with routers by computing signatures that represent spam sources, thus propagating periodical updates. Fingerprinting capabilities are required on both the SMTP servers, as well as the participating routers. The risk of overloading the high throughput resources (*such as routers*) does exist, and furthermore, the degree of intelligence that can be applied at this point to ensure respectable throughput, including ensuring zero false positives, is limited.

2.4 Security and Privacy Considerations

Email is considered mostly a means of personal communication in the context of participants' selectiveness. Therefore, security and privacy aspects are fundamentally important. The next paragraphs provides a discussion of spam filtering work which considers this particular perspective.

2.4.1 Security Aspects

An efficient way of hiding a spammer's identity and the spam's source of origin is by hijacking computing resources. These resources, which do not belong to the spammer, are subsequently employed to perform spamming operations, and thus, appear as spam sources themselves. This can be done either on a transient basis or for longer periods without the legitimate resource owner's knowledge. Botnets remain a real nuisance when dealing with spam proliferation due to the continued increase in hijacking sophistication, distribution, and execution capabilities. Various studies on botnets have been conducted, researching their behaviours, characteristics, and methods of dealing with them. In [113] the authors provide an interesting perspective describing Trinity. Based on P2P-architecture and implemented as a SpamAssassin plugin [114], Trinity is founded on the premise that automated botnets send a large amount of unsolicited email in a relatively short period of time. Participating peers process and provide information related to the mail relays they are associated with and have information about. This exchange of information is used to measure associated email sending rates. Another detailed study on characterizing botnets, provided in [115], where the authors present a framework designated AutoRe intended to provide a signature generation framework for identifying botnet sourced spam. Similar work on a malware generated email identification method using clustering algorithms (*based on a modified Levenshtein distance scheme and combined with Jaccard similarity coefficient*s) is described in [116].

To mitigate source counterfeiting and introduce additional traceability elements from the sender's end, email authentication [117] has been researched and put on the research map [118] [119]. Numerous efforts to this extent can be identified—notably, the Sender ID Framework (*SIDF*) and the Domain Keys Identified Email (*DKIE*). However, they do not seem to have reached the critical mass as originally expected. [118] argues that large enterprises with substantial numbers of email addresses may face challenges when trying to adopt such approaches. In addition to the concern of potentially breaking typical mail forwarding capabilities, SIDF is also prone to having spammers registering as legitimate users.

DKIE is heavily based on Public Key Cryptography. In [120], the authors discuss how Secure/Multipurpose Internet Mail Extensions (S/MIME), which is employed by DKIE, can be improved. Ironically, some argue that spammers themselves are among the more avid adopters of this approach [121] - where authentication is applied in isolation, spammers endorsing this technique have a golden key for sending unsolicited bulk email. Another pre-send filtering technique intended to stop email at the source rather than trying to control it at the recipient end is presented in [116]. The basic concept is based on the ability to ensure the legitimacy of the source. Email that is not verified using an e-stamp is forwarded for spam scrutiny. Here, the e-stamping scheme is based on IAPP, an integrated authentication process platform and it is employed in conjunction with dynamic black or white listing, behavioural identification, and Bayesian techniques. Conceptually, the originator of the email marks the source with an e-stamp (*constituted of a number of specific properties*) by interacting with the IAPP, which provides the necessary stamp information. Subsequently, the email verification process will check the validity of the stamp and submit the email to the mail filtration process. The recipient's end email service updates its dynamic lists according to the outcomes of the mail filtration scheme. This approach has numerous benefits, including the ability to stop illegitimate email from being sent by establishing the presence or lack of an adequate IAPP stamp. It also ensures that the communication exchange is verified email and truly legitimate. The challenge here is related to the degree of intrusiveness the mail user is willing to forego in return for added security and legitimacy. Furthermore, the IAPP functionality introduces additional operational overhead, as well as complexity from a service provisioning perspective.

In [122], the authors employ email authentication to ensure sending legitimacy. An authentication agent interacts with a user profile that is governed by typical environment security parameters at the operating system level, such as login and password. This is processed by the mail client via a faithful sender agent, which then determines user identity. Broadly speaking, challenge response techniques such as this one rely on the presentation of a challenge, such as a password, and expect an associated response, which is computed in real or predefined time. Various commercial products employ some sort of challenge/response features [123], and an interesting pre-sending approach based on these principles is presented by [48]. The sender retrieves the recipient's email and is solicited to interactively supply a reply to a respective challenge before the email is actually sent. This only happens the first time that an email exchange transacts between specific senders and recipients.

The Completely Automatic Public Turing test to tell Computer and Humans Apart (*CAPTCHA*) [124] is similar to general challenge response approaches. CAPTCHA is an elegant technique intended to ensure a degree of confidence that interaction with a specified service is actually being performed by a human being. This degree of confidence depends on the level of logic quality it is based on. The work presented in [125] and [126] propose schemes based on this approach. In [125], the email user is expected to present specific information to perform the required authentication before being able to send email. The authentication protocol implemented in SASL (*Simple Authentication and Security Layer*) [127] introduces additional steps within the email sending process, making it increasingly difficult for spammers to fully automate the act of sending emails in any straight forward fashion. The challenge with this approach is the increased intrusiveness of sending email from an end user perspective. The approach described in [128] employs additional features, including the application of server side certificates for email servers authenticity. A hash casting technique introduces a stamp intended to identify the validity of the sender's origin and intent by exploiting processing power at the sender's end to generate the stamp. The stamp can be based on a number of sources, for example, SHA-1 [128]. The processing delay introduces a step which renders sending automated mass email much less efficient.

While CAPTCHA based schemes have become increasingly widespread, challenges remain with respect to the level and degree of intrusiveness such approaches creates for the end user. Similarly, challenge response based approaches have become increasingly more popular but are faced with a number of comparable challenges, including overall speed degradation of the communication exchange and deadlocking. The latter occurs when both ends of the communication path are based on challenge responses. To address these challenges, the work presented in [129] proposes an innovative approach: the Mail Transfer Agent (*MTA*) maintains a trusted list associated with each email user, and when an email's source is not within this trusted list, the sender is challenged. If there is a subsequent reply, the sender is newly added to the recipient's trusted list. To tackle the deadlock issue, the authors propose a scheme where the sender sends an email to a recipient; the recipient's address is then automatically added to the sender's trusted list. Additionally, respective trusted lists of both senders and recipients can be modified manually by administrators. This approach involves an extension to the SMTP protocol, intended to facilitate the proposed scheme, as well as targets the tempo challenge associated with typical approaches. This introduces complex challenges associated with the protocol itself, which is a widely adopted standard and entrenched in today's email infrastructure. Any change to it would therefore require considerable effort to ensure the required degree of compatibility. SpamCooker is also an example that employs a similar approach [130]. This technique considerably disturbs spammers' ability to effectively employ traditional techniques for sending unsolicited bulk email. While this is an area which is avidly researched, various studies show that such approaches can still become victims of a number of attacks, which may either totally bypass or considerably reduce overall effectiveness, depending on the original quality and strength behind the adopted implementations [131][132].

Having a cost element attributed to the process of sending email will also harm the spammer's business model, rendering it less favourable. In [52], the authors argue that this approach is crucial to stopping spam at the source and that privacy related issues can be less amplified with this method. LCP (*lightweight currency protocol*) [3] and micro-payments [3] [133] [48] are typical applications, although concerns regarding potential security issues [134] have been raised. Additionally, there is the complexity of having the necessary logistics to support such an approach, especially when dealing with differences in policies that may exist between service providing organizations [3]. The potential variety of platforms may also need to be considered.

2.4.2 Personalization and Privacy

It would be beneficial to briefly consider the end user's perspective of spam and the challenges it gives rise to. One dimension, as already indicated, is that there is an evident element of personal subjectivity: that is, what is considered spam and what is not. Additionally, the level of understanding regarding spam (*including how it can be alleviated and what tools and techniques can be employed*) varies considerably across the entire email user base. This is not surprising, given the number of email users worldwide [135], amplifying the importance of putting the end user at the centre of the stage. Therefore, filtering approaches and solutions should seek to be as unobtrusive as possible, to ensure a uniform user experience while allowing for maximum exploitation of spam reducing opportunities [136].

Different recipients may use different measures to segregate spam from valid email. It is therefore a challenge to ensure that there is a relevant degree of influence from the user to ensure a level of personalization while automating the classification process to the maximum extent possible. This presents a continuing juggling act between the accuracy that can be provided by a small, personal spam reference base set, and more generalized classification provided by a larger base set, which could potentially lose out on the personalization aspect, assuming both perspectives are considered in isolation. Keeping the personalization aspect as non-intrusive as possible is another challenge, as having the end user manually tag email on a sustained basis (*to improve accuracy based on personal considerations*) incurs a process overhead that will annoy a number of users.

In [137], the authors present an approach that specifically considers personalization in a non-intrusive fashion. Based on statistical methods, they employ a more generalized training set for initial setup, which subsequently adapts itself to the end user's context of spam. In the presented perspective however, the time required for rebuilding statistical models and the sizes of the filters, proves challenging. These process demand significant processing and memory requirements on the ESP's end and can have a considerable impact where there are a substantial number of relatively large filters and a large number of users. This is one of the reasons why other approaches typically utilize the resources available at the user-level, which is the Mail User Agent (*MUA*) end [6].

In [138], the authors also present work on the personal element in spam filtering, employing Quickfix, a proposal using a Vector Space Search (*VSS*), or a word frequency comparison method similar to whitelisting approaches. This is employed to compare incoming mail with a localized spam word dictionary. Vector space search is relatively simple and quick, resulting in limited intrusiveness in terms of the entire mail processing and sending operation. Subsequent to the VSS operation, if the outcome is marked as spam, the message is forwarded to a Naive Bayesian process. If the outcome of this second operation is still spam, the spam word dictionary is updated accordingly. The authors conclude that while neither VSS nor Bayesian provide an adequate level of consistency in isolation their combination provides an effective approach.

Privacy is another issue that affects the application of spam filtering approaches, which scrutinizes email content to establish email legitimacy. This is further amplified when individual third parties other than the sender, intended recipients, and respective service providers are involved. In [139], the authors discuss the ALPACAS framework, an approach based on a fingerprinting scheme as a solution. ALPACAS guarantees intactness of the email features while forwarding only a subset of mail content to participating agents for spam classification, rather than a full mail dataset. As a result, the privacy of email can be ensured.

2.5 Discussions and Observations

This chapter examined a number of approaches to spam filtering, consulting and reviewing a number of sources. Each source varied in scope and covered various aspects of spam filtering techniques.

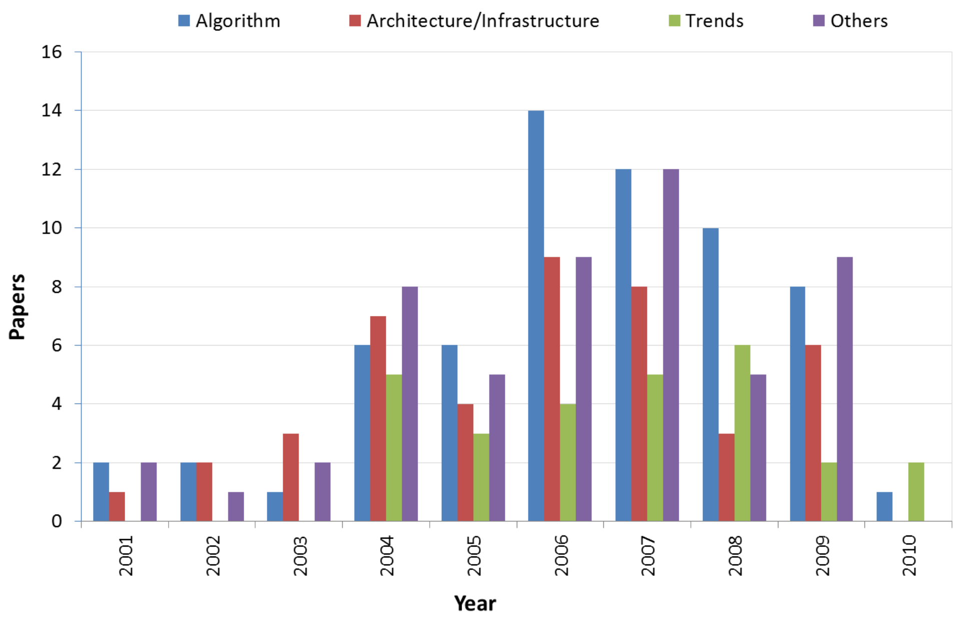


Figure 2-3: A classification of surveyed papers

Figure 2-3 shows the number of papers surveyed for the part of this research, totalling 102 papers ranging from 2001 and 2010 and categorized according to focus, such as Algorithm, Architecture / Infrastructure, Trends, and Other. This does not in any way imply that these are mutually exclusive standpoints- rather the key (*and not necessary sole*) focus of the research work surveyed.

Work classified under *Algorithm* reflects research that primarily discusses types of classification schemes and associated algorithms, including machine or non-machine learning approaches, such as Bayesian, Heuristics, etc.

Architecture/Infrastructure focused work is principally concerned with the design and implementation of spam filtering infrastructures - traditional as well as emergent.

Work classified under *Trends* refers to discussions focused on how spam filtering approaches change over time, which includes the consideration of emerging methods.

Other type of research is categorized under the '*Other'* category - the works reviewed that could not be directly classified under the other established categories are categorized under this class, which includes legal and organizational perspectives, for example.

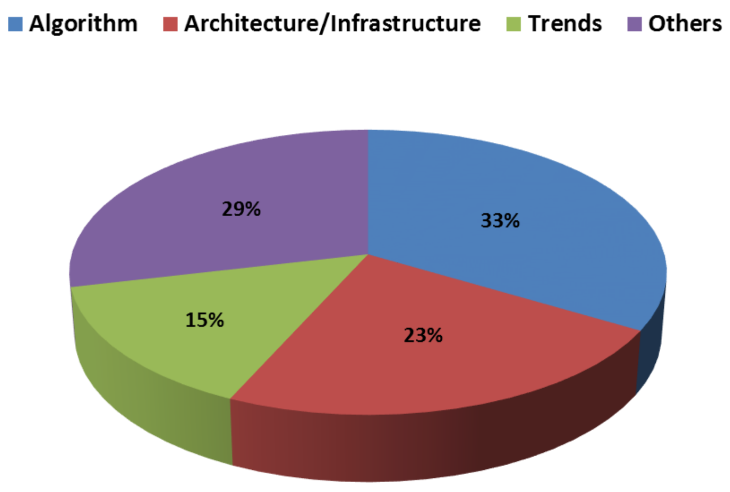


Figure 2-4: The focus distribution of the surveyed papers.

Figure 2-4 indicates that 33% of surveyed papers focused primarily on the algorithmic perspective. Architectural dimensions were considered in 23%, while about 15% reflected on general trends in anti-spam.

Table 2-1 summarizes a number of key perspectives that influence the overall value proposition of specific spam filtering technologies, approaches, and techniques. The aspects compared, namely perspective and technique, are referred to as pre or post sending filtering approaches. They are also associated with the level of complexity involved, overall performance, and quality, as well as how popular they are, in terms of reach. Table 2-1 also assigns an identifier to each dimension demonstrating whether the approach employs machine learning (*ML*) techniques or otherwise (*Non-ML*). The table also shows whether distributed and collaborative elements are in use. The last two columns in Table 2-1 illustrates the levels of intrusiveness and privacy preserving from an end user perspective.

Table 2-1: Surveyed work perspectives

Perspective	**Technique**	**Pre-Send**	**Post-Send**	**Complexity**	**Performance**	**Basic Quality**	**Reach**	**ML**	**Non-ML**	**Distributed**	**Collaborative**	**Privacy**	**Intrusiveness**
Algorithm	Rule Based	✓	✓	⇩	⇩	⬄	⇧	×	✓	○	○	⇩	⇩
Algorithm	Signature	✓	✓	⇩	⬄	⬄	⇧	×	✓	○	○	⇩	⇩
Algorithm	Bayesian	×	✓	⬄	⬄	⇧	⇧	✓	×	○	○	⬄	⬄
Algorithm	kNN	×	✓	⬄	⬄	⬄	⬄	✓	×	○	○	⬄	⬄
Algorithm	ANN	×	✓	⬄	⬄	⬄	⬄	✓	×	○	○	⬄	⬄
Algorithm	SVM	×	✓	⇧	⇧	⇧	⇩	✓	×	○	○	⬄	⬄
Method	Monetary	✓	×	⬄	⇧	⇧	⇩	○	○	○	○	⇧	⇧
Method	Chall./Resp.	✓	×	⬄	⇧	⇧	⇩	○	○	○	○	⇧	⇧
Method	Semantic	×	✓	⬄	⬄	⬄	⇩	○	○	✓	✓	⇩	⬄
Method	Social Net.	×	✓	⬄	⬄	⬄	⇩	○	○	✓	✓	⇩	⬄
Arch./Infra.	Centralised	✓	✓	⇧	⬄	⬄	⇩	○	○	○	○	⇧	⇩
Arch./Infra.	GRID	×	✓	⬄	⬄	⇧	⇩	○	○	✓	✓	⇩	⬄
Arch./Infra.	P2P	×	✓	⬄	⬄	⇧	⬄	○	○	✓	✓	⇩	⬄

Where:

✓ - Represents yes
× - Represents no
⇩ - Represents low
⇧ - Represents high
⬄ - Represents average
○ - Represents an element not context

E.g. - The Rule Based Technique is a Non-ML approach, can be applied both from a pre and post send perspective, low complexity to implement. It is considered a medium level of quality in terms of spam filtering application. It is considered a non-intrusive approach, yet one which is not very privacy aware.

Rule, signature, and Bayesian approaches have been widely discussed, and the overall, long-term value of the latter approach tends to be better than the first two. However, while signature and rule based approaches are generally considered less intrusive, Bayesian based classification remains highly popular [140][141][142][143]. While performance and requirements vary considerably, scalability is an on-going concern for a number of traditional Bayesian based approaches, by virtue of their dependence on memory availability and performance [129]. Spammers can also learn which terms influence Bayesian filtering outcomes and adopt an approach based on the additional insertion of spurious content that is intended to influence overall results and outcomes. This is accomplished by weakening the effect of any high order ranked tokens used to infer whether the content is spam.

Other common machine learning approaches include Artificial Neural Networks (*ANN*), k Nearest Neighbour (*kNN*), and more recently Support Vector Machines (*SVMs*). kNN approaches are normally subjective to noise, indicating that errors in the training set can easily induce misclassification. They also tend to be computationally intensive in terms of larger datasets. Similarly, whilst the accuracy of ANN classification is high, this approach has a tendency to require significant computing time for spam classification [133]. In online anti-spam classification environments, the balance between having an up-to-date training dataset and resources available to train or retrain the artificial neural network is critical. This requires a continued effort trying to strike a balance between functionality and effectiveness. It is common to identify real-world scenarios where ANN based approaches are used in conjunction with additional filtering schemes rather than in isolation. SVM approaches have shown their effectiveness in spam filtering. Compared with Bayesian approaches, SVM approaches are even more computationally expensive, which can constrain their maximum potential application in online implementations.

Filtering approaches, although not as popular as machine learning approaches, are based on challenge response, authentication, and CAPTCHA schemes. These tend to provide a higher degree of performance in terms of the ability to mitigate spam proliferation. They also ensure a relatively higher degree of privacy and security but can be more intrusive overall from a user experience perspective. From a holistic architectural perspective, numerous methods in application have been identified. For the purposes of simplicity, these can be grouped under a number of umbrella areas, namely Mail Transfer Agent (*MTA*), Mail User Agent (*MUA*), Hosted, P2P, and grid computing based approaches. In MTA based schemes, the actual implementation is commonly identified as either an extension to the MTA or part and parcel of the package. The MUA approach commonly involves rules which can be programmed within the MUA environment itself. They also include plugins with filtering intelligence, as well as external solutions, which work outside the main MUA. SMTP proxy services, usually available as separate standalone logic, are also popular. Under normal circumstances, the hosted approach does not involve any (*or perhaps minimal*) intelligence from the MUA or MTA perspective, as the entire (*or most of the*) anti-spam filtering service is performed elsewhere.

P2P, and grid computing based paradigms, where participating nodes are able to share various resources such as storage, processing power, and connectivity in a collaborative way, are considered as emerging, high-performance spam filtering schemes. High resiliency is a key advantage of such architectures, but their subjectivity to participating nodes with malicious intent is of concern. Another benefit of these approaches is their ability to share spam intelligence relatively easily, widening the scope for increased collaboration. The same applies to social network and semantic web based approaches, which are also garnering increased attention. These emerging approaches, however, are not currently widespread compared with traditional approaches.

Other approaches exist as well. Worth mentioning at this stage is that specialized devices that are positioned strategically within an enterprise network to minimize spam influx are increasingly common. Such devices include intrusion and prevention detection systems (*IPD/S*), network devices which are either enabled out-of-the-box with specific spam filtering functionality or which can be extended to enable such functionality, as well as specialized devices specifically and solely intended to act as spam filtering devices. Table 2-2 provides a market snapshot of popular solutions to spam filtering. The Product column describes the product name whilst the second column (*Spam Filtering Type*) shows the type of implementation and approach adopted.

Table 2-2: Typical products and types

Product	Spam Filtering Type
Google / Yahoo / Hotmail	Hosted
McAffee / SpamKiller	Appliance

NetIQ / BrightMail / GFI Essentials	Centralized Gateway
NetworkBox	IPS/IDS
IronPort / Barracuda / SpamTitan	Appliance
BitDefender	MTA Extension
SpamAssassin	Software
Cloudmark SafetyBar / Spam Bayes / Outclass	MUA Plug-in

Rather unsurprisingly, it transpires that no one approach seems to fit all requirements. In the context of the methods, algorithms, and architectures employed to mitigate the spam challenge, the selection and combination of the techniques discussed have varying outcomes that influence their overall success in terms of adoption and implementation. In isolation, performance figures such as speed and accuracy are not necessarily inductive to the most effective approaches in the real-world. Architecture brilliance doesn't necessarily create the most effective environment overall for spam filtering. The consideration of other important factors comes into play, including the complexity of implementation, end user intrusiveness, and privacy amongst others.

2.6 Summary

The literature survey presented in this chapter focused on emerging approaches to spam filtering. Based on the literature reviewed, centralised anti-spam architectures tend to be more apt for streamlined management simplicity and non-intrusiveness. This in conjunction with the capitalization of resources at the client's end which can improve the scope for increasing anti-spam intelligence [139][34][76][144][113]. Combinations of filtering schemes are considered to provide better overall results when compared with singular approaches [145]. Emerging approaches based on grid computing, P2P computing, semantic web and social networks bring increased opportunities in terms of scalability, formalization of spam definitions and personalization, as well as collaboration between the participating parties by sharing resources as well as increasing spam intelligence.

It is still safe to state that a number of modern day email service providers still rely primarily on centralized, services side based architectures. These approaches typically use traditional classification schemes for most of their spam filtering efforts, whilst striving to ensure an 'adequate' level of personalization. The costs associated with serving a large population of end users using a rigid centralized function cannot be however ignored. The elements associated with costs do not solely concern the financial standpoint, but encompass aspects including classification scheme performance and requirements, storage and network perspectives. Fluctuation of user population, both in terms of concurrency as well as their number also have a considerable impact.

Spammers are getting smarter and continuously trying to come up with approaches that enable them to circumnavigate spam filtering schemes. Active research in emerging approaches shows that there is a continued effort trying to come up with better and alternative anti-spam schemes. Given that the operational landscape of spam ecosystems is continuously changing, different techniques that can be applied to old and new problem areas are being sought. Social networks have become a major breeding ground for spam related activities. Research in these specific areas including the behavioural model space [146] [107] has also shown to have been intensified with continued advancement. Considering the scale of the spam challenge, this survey work also pointed out that distributed computing paradigms can be employed as an enabling technology for high performance machine learning approaches to spam filtering. This rationale is considered as a baseline proposition for further research towards the feasibility of a flexible, collaborative, and distributed spam filtering architecture.

Finally, the quality as well as the number of related literature demonstrates the effort as well as the significant advancement that has been, is being and continues to be made with respect to coming up with better and alternative schemes for tackling the spam challenge.

CHAPTER 3 – MapReduce based Distributed SVM for High Throughput Spam Filtering

As indicated earlier, machine learning techniques such as ANN, Naive Bayesian, Decision Trees and SVM have been applied in spam filtering [4] [9] [17] [10]. Specifically, SVM based approaches have persistently gained popularity in terms of their application for text classification and machine learning [4] [9]. This popularity can be attributed to the sound theoretical foundations as well as good performance. However, SVM's Achilles heel has consistently been related to real world performance, especially in the context of large training sets.

In this chapter, an SVM based algorithm, able to capitalize on a typical distributed computing framework is designed, presented and performance evaluated in a number of different constructs. The algorithm is intended to be able to reduce spam filter training time significantly whilst ensuring the required degree of accuracy. Shorter training times provides the opportunity to re-train the machine learning model frequently. This ensures that classifier currency is kept high.

The rest of chapter 3 is organized as follows. Section 3.1 presents a discussion on related work. Section 3.2 provides an overview of support vector machines. Section 3.3 introduces the MapReduce framework using Hadoop as a reference implementation. Section 3.4 presents the design and implementation of MRSMO, a distributed SVM. Section 3.5 describes a set of comparisons between the original sequential and proposed distributed versions of the SMO. These include performance, accuracy and scalability. Simulation results using HSIM - a Hadoop MapReduce Simulator - are also presented, alongside a comparison with an MPI based approach (section 3.6) and section 3.7 provides a summary of this chapter.

3.1 Related Work

Spam filtering takes many shapes and forms, and is continuously evolving [147]. MapReduce is employed in a variety of areas ranging from large scale data analysis, search optimization, machine learning, forecasting and social media [148]. To date, MapReduce primary application is focused towards data intensive tasks rather than for computation in general. However, various efforts to this extent exist as well, including the work presented in [32][149][34]. The application of M/R in spam filtering is also common within the industry by the likes of Microsoft, Yahoo, Google [150].

The traditional approach for SVM training is a batch process with off-line learning. Online SVM training on the other hand introduces the notion of incremental updates. A key difference between the two approaches is that whilst the former normally provides finer grained control over classification tolerances, the latter has lower training time requirements at the outset.

As already indicated, SVM training is a computationally intensive process. In an effort to increase performance, scalability, reduce complexity as well as ensure that the required level of classification accuracy is retained, numerous avenues have been explored. Such techniques include optimized SVM kernels, decomposition techniques as well as the application of distributed computing techniques.

SVM's rely on the number of established support vectors for classification. Support Vectors are those which lie near the hyperplane and which separates the cluster of vectors, introducing the largest margin between ham and spam. To improve performance, Li et al try to decrease overall training complexity and classifier categorization by reducing the number of support vectors [17]. This is achieved by tuning the respective Lagrange [151] multipliers accordingly. Tuning is based on the assumption that linearly dependent support vectors are not required for the final decision function.

Sculley and Wachman present an online approach which tries to avoid re-training [21]. This is achieved by adopting an incremental approach using new training samples as they become available. Newly acquired training samples are used in conjunction with the current (*last known*) hypothesis as a baseline when a poorly classified sample is identified. This approach is congruent with a typical anti-spam scenario because it limits re-training to a need basis whilst still keeping up-to-date. The authors also argue that trying to maximize the SVM's separating margin in real-world application scenarios may not be necessary. In this regard, the work suggests the relaxation of a number of key SVM parameterization aspects.

Only the last *n* samples are employed as a baseline for optimization for example. This alleviates the issue of a '*growing*' training input size but will also influence the optimization scope. Such a change will however increase accuracy related challenges. The experimental results presented indicate that this is not significant and may not warrant the cost associated with the full consideration of the training set. This work also presents a number of comparisons with respect to reducing optimization problem size, iterations and updates.

Lun et al. [17] also try to decrease overall training complexity and classifier categorization by reducing the number of support vectors. Poulet also presents an adapted LSVM formulation for SVM training intended to reduce overall complexity [152]. In this effort however, data chunking, or rather splitting the training data is applied. This aspect is similar in part to the approach adopted in this dissertation work. However, reliance on specialized coding techniques and supporting libraries, rather than adopting widely available and commodity approaches is considered to be a disadvantage.

Training data is commonly represented as a set of input vectors, treated equally in terms of overall importance. Li et al. extend the basic SVM notion by introducing a weighting element to each class of email (*spam or ham*) [13]. An additional enhancement is achieved by assigning training vectors with different weights. The technique is intended to mitigate problems associated with imbalance. Imbalance refers to the value of each individual training element (*instance*) in relationship with the other elements within the training set. This can increase real world classification accuracy.

Accuracy is also influenced by false positives. False positives have a higher associated cost in comparison to spam which is not identified as such. In an attempt to reduce false positives, a number of SVM approaches have been studied and proposed, [9][69]. False positives are commonly the result of limited tolerance thresholds during classification. This result in an increased spam capture count but also increases the number of emails classified as spam when they are not.

Chiu and Huang employ a combination of SVM and Naïve Bayesian classification in their work [153]. Another multiple classifier based approach is discussed in [154], where the authors argue that simple email body content may not contain enough information for classification. In the proposed approach, an initial phase filters spam mail using more definite information such as source and typical keywords. Hyperlinks within the email body are subsequently utilized to extract additional relevant features which are employed in the second, SVM phase of the process. This phase is employed for features which are considered more subjective and intended to minimize classification inaccuracy.

Zanghirati and Zanni [26] propose a decomposition technique for increasing performance and scalability. This takes the form of splitting the overall SVM quadratic programming challenge into a number of smaller sub-problems, similar in part to this work. Individual results of each separate computation are then subsequently combined. However, the caching strategy of the approach heavily influences the levels of re-computation required for SVM convergence. Cao et al. [18] present another parallel SVM approach using MPI. A comparable approach is adopted by Woodsend and Gondzio [151]. Here optimization is achieved by removing the dense Hessian matrix and matrix partitioning for better data locality. Whilst performance improvement can be achieved by MPI based parallelization, these approaches tend to suffer from poor scalability when heterogeneous computing environments are employed [155][156].

Do et al. [157] present an innovative variation by offloading core processing elements to a GPU (*graphics processing unit*). The experimental results show remarkable speed improvement when compared with traditional CPU based computation. The application of the GPU is also discussed and featured in the work presented in [19]. Here the authors consider a M/R based approach exploiting the multi-threading capabilities of graphics processors. The results show a decrease in processing time requirements on the order of 9 to 35 times. A key challenge with such approaches lies in the specialized environments and configuration requirements.

Capitalizing natively on the multi-core capabilities of modern day processors, Chu et al. [158] evaluate the performance of a number of algorithms including a M/R based SVM. Dual-core processor design removes most of the communication overhead incurred in distributed processing scenarios. The authors argue that this approach is more pragmatic for real world applications when compared with specialized implementations such as the one presented in [159] where a cascaded SVM scheme is presented. Here, each training data set is used as input for each SVM instance and an iterative re-training (*cascading*) process is applied. The effectiveness of such an approach relies primarily on the number of cascade iterations performed. Where substantial iterations are required, the overall effectiveness can be reduced because the time required to combine the respective support vector sets cannot be discounted. This issue is also highlighted in [156].

The consideration for personal and privacy elements in spam classification presents a challenge for shared training data sets. To mitigate this challenge, Xu and Zhou [160] base their work on a strategy which includes test samples concurrently with the training variants. Based on inference, the discussed Transductive SVM is intended to identify specific classification parameters for specific sets of input (*mail*). This is in contrast to solely relying on a singular training set. In this work, the authors take into consideration personal mailboxes in conjunction with global spam corpora to improve overall classification performance. The personal mailbox based training data is unlabelled in contrast with the labelled global spam corpora variant. The ability to classify spam without looking at the content of email also ensures increased privacy in spam filtering. This can be achieved by selecting a feature set based on attributes which exclude content or derivatives. Along the same lines, Hershkop and Stolfo [23] study the applicability of using the address, domain, recipient number and size characteristics as features for SVM training.

Image based spam has been continuously on the rise. The same tendency applies for increased varieties of spam which include content beyond simple text. Here, the spam payload can be of a significant order of magnitude larger than simpler text based equivalents. Typical issues related to bandwidth and processing requirements for spam classification and filtering are subsequently amplified. Research to this extent has garnered considerable attention and forms the basis of numerous works including those presented by [161] [162] [163].

Zou et al. present a pyramid match SVM kernel employing image histogram counts [164]. On the other hand, Mehta et al. employ image clustering using features such as colour, texture and shape of spam image as input to the SVM classifier [165]. A further increase in accuracy reaching 98.5% is achieved, but with a relatively small number of training samples.

In [81] anti-spam agents collaboratively contribute towards the classification of spam. This is done by generating and processing representative email hash values (*signatures*) from respective email services. Spam signatures are persisted in localized databases. These databases include signatures generated for locally processed email designated as spam as well as those exchanged by other participating agents. Should initial signature comparison not provide the required degree of confidence that the email is spam, subsequent SVM processing takes place. Spam tagged mail will have the respective signatures propagated to the other participants for immediate subsequent identification. The ability to employ numerous distributed computing resources concurrently increases performance and scalability potential substantially. However, the implementation suffers from elevated network traffic burden which is introduced during hash exchange cycles. The approach is also prone to denial of service attacks by instigating massive propagations of invalid or malicious signatures. In [21], the authors discuss how the signature exchange issue can be mitigated by reducing the number of updates and the problem domain size.

3.2 The Support Vector Machine

Various machine learning approaches exist and which can be adopted for spam filtering. In general, spam filtering can be considered as a text classification problem – without ignoring the evolving forms of payload and delivery approaches, including images and attachments. Support Vector Machines (SVMs) have been proven as a statistically robust machine learning method [15] which yields state-of-the art performance on general text classification [166], [11], [14], [15], [24]. Different kernel functions can also be employed for the actual decision function. Furthermore, SVMs have very good generalization performance. These attributes form the underlying motives for the preference and adoption of SVMs as a baseline supervised machine learning classifier for this research work, especially in a construct towards ensuring that the variety and velocity perspectives of the spam challenge are tackled.

SVM classification is founded on the notion of ‘hyperplanes’ [24]. The ‘hyperplanes’ act as class segregators, such as spam or ham in the context of mail classification. Figure 3-1 shows an example set of SVM hyperplanes. H_1 does not separate the respective classes. H_2 does, however it does not introduce the widest separating margin, which is the case for H_3.

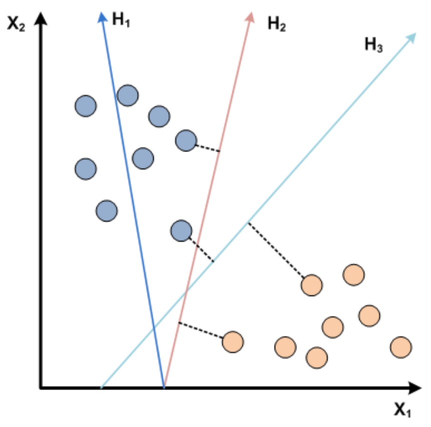


Figure 3-1: SVM (Linear) separating hyperplanes

The identification of the support vectors forms the basis of the decision function. In a linear separation (*of classes*) construct, the separation hyperplanes via Equation 3-1 and pictorially in Figure 3-2:

$$\boldsymbol{x_i} \cdot \boldsymbol{w} + b \;\; +1 \; when \; y_i \; = \; \mathbf{+1}$$
$$\boldsymbol{x_i} \cdot \boldsymbol{w} + b \leq -1 \; when \; y_i \; = \; \mathbf{-1}$$

[3-1]

Where H_1 and H_2 are the planes $x_i \cdot w + b = +1$ and $x_i \cdot w + b = -1$ respectively. The points on the planes are the support vectors and the separating hyperplane margin is defined by $d^+ + d^-$, where these describe the shortest distances to the closest positive and negative points respectively. The distance between H_1 and H_2 can be described via $2/\|w\|$.

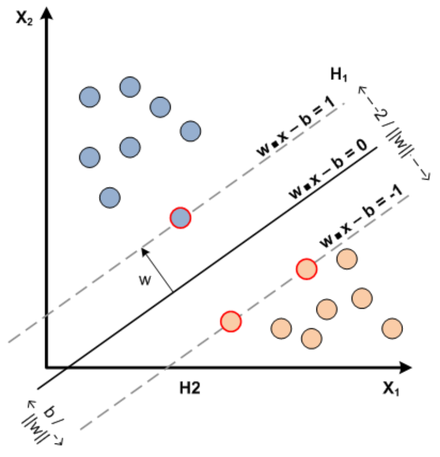


Figure 3-2: SVM (Linear) decision function

To maximise the margin between H_1 and H_2, $||w||$ needs to be minimized – a constrained optimization problem solved using a Lagrangian multiplier method described by:

$\underline{\text{minimize}}$ $||w||$, subject to boundary condition, i.e. $\min f(x)$ s.t. $g(x)=0$
where f: $1/2\ ||w||\hat{}2$ and g: $[y_i(x_i \cdot w - b] - 1 = 0$. **[3-2]**

Where linear separation is not possible, data is mapped in higher dimensional space, via '*kernels*', to establish linear separation, i.e.: $x \mapsto \{x_2, x\}$. Kernel examples include Polynomial $[K(x,y) = (x \cdot y + 1)^p]$ and Radial Basis Function (Gaussians) $[K(x,y) = \exp\{-||x-y||^2 / 2\sigma^2\}]$.

3.3 MapReduce

Various shapes and forms of distributed and parallel computing metaphors, frameworks and API's exist. These include Condor/MPI Condor, Open MP, Sun Grid Engine, GridGain, Beowulf and MapReduce amongst others. Popularized by Google and initially focused on high performance, Internet scale, search related challenges, MapReduce has gained a lot of popularity due to its simplicity, scalability and accessibility. The out of the box scalability, relative simplicity, fault tolerance as well as a solid distributed architecture are believed to make it a good candidate for an high throughput infrastructure for typical text classification challenges associated with spam.

MapReduce is a parallel and distributed programming model supporting data intensive applications. Programmatically inspired from functional programming, at its core there are two primary features, namely a *map* and a *reduce* operation. From a logical perspective, all data is treated as a Key (K), Value (V) pair. Multiple mappers and reducers can be employed. At an atomic level a map operation takes a {K1, V1} pair and emits an intermediate list of {K2, V2} pairs. A reduce operation takes all values represented by the same key in the intermediate list and generates a final list. Whilst the execution of reduce operations cannot start before the respective map counterparts are finished, all map and reduce operations run independently in parallel. Each map function executes in parallel emitting respective values from associated input. Similarly, each reducer processes different keys independently and concurrently.

3.4 The Design of MRSMO

The scalability of SVMs in data training still remains an open question by virtue of the involvement of constrained convex quadratic programming challenge associated with the dense Hessian Matrix involved during optimization. In simpler terms this refers to the computing time required which is quadratic to the number of training data elements.

Numerous SVM formulations, solvers and architectures for improving SVM performance have been explored and proposed [159], [167] including Message Passing Interface (*MPI*) based parallel approaches [152][168][15]. SVM decomposition is another technique for improving SVM training performance [11][25][169]. A widespread practice is to split the training data and use a number of SVMs to process the individual data chunks. This approach typically splits the training data set into a number of smaller fragments. This in turn reduces the individual training set size and subsequently the overall training time. Most forms of decomposition which are based on a data splitting strategy approach tend to suffer from issues including convergence and accuracy. The latter because typical SVM training is a global optimization problem which typically relies on the entire dataset to infer the final objective function. Challenges related to chunk aliasing, outlier accumulation and data imbalance/distribution tend to intensify problems in a distributed SVM context. Again, this in turn impacts generalization, accuracy and convergence. Techniques such as random sampling have also been shown to exhibit similar accuracy challenges because of the induced probability distribution variance [170].

On the other hand, selective sampling techniques applied to SVMs try to sample the training data intelligently to maximize the performance of SVMs. However these normally require many scans of the entire data set which incur high computation overheads. Consequently, the scalability of SVM in dealing with large data sets still remains a challenge. The motivation behind this part of the dissertation work is to establish an efficient, yet simple, distributed SVM algorithm based on the highly scalable MapReduce framework [149] for large scale SVM training. The distributed SVM proposed herewith is built on the Sequential Minimal Optimization (SMO) algorithm [11] and implemented for MapReduce. Compared with MPI, the MapReduce framework is highly scalable in processing large data sets and it can handle node failures which are critical for long running jobs. An abstract representation of this approach is illustrated below.

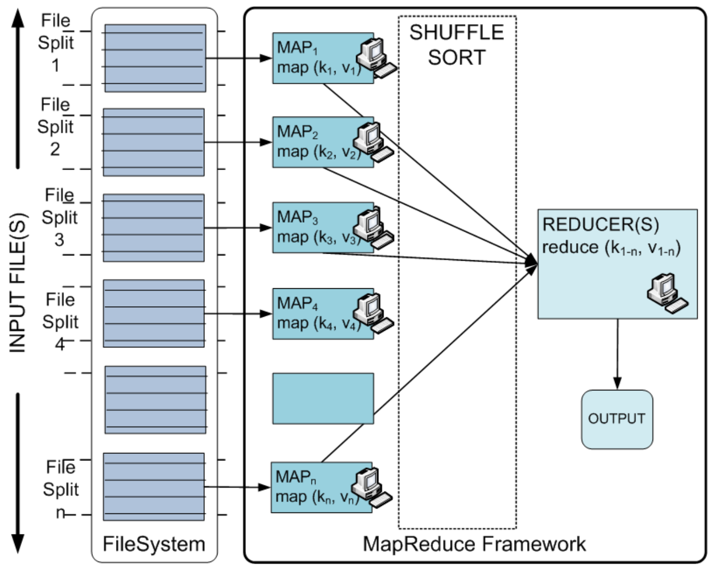


Figure 3-3: MapReduce Architecture

3.4.1 Sequential SMO

There are numerous SVM algorithm variations and implementations, including SvmLight [171] LibSVM [172] and LASVM [173]. Typical SVM training is challenged by memory and computational requirements especially in the context of large training corpora [174]. One of the more popular SVM solvers is the Sequential Minimal Optimization (*SMO*) algorithm by virtue of its simplicity and overall effectiveness.

The SMO algorithm tries to minimize the complexity in computation by eliminating the need for an iterative quadratic optimizer [15]. Recalling the basic SVM classifier which takes the form of f(x) = w T_x + b, as applied to ham from spam separation and which is representative of binary classification, the prediction takes the form of y =1 for f(x) ≥ 0 and y = -1 if f(x) < 0 (y is the class label, ham or spam respectively). The dual representation (*Lagrange formation*) can be formulated as Equation 3-3.

$$f(x) = \sum_{i=1}^{m} \alpha_i \, y^{(i)} \left\langle x^{(i)}, x \right\rangle + b \qquad \textbf{[3-3]}$$

The basic SMO algorithm adopts the approach formulated in equations 3-4, 3-5, and 3-6 towards minimizing computing resource requirements in terms of solving the SVM optimization problem:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)}\ y^{(j)} \alpha_i\ \alpha_j \langle x^{(i)}, x^{(j)} \rangle \tag{3-4}$$

$$\text{subject to } 0 \leq \alpha_i \leq C,\ i=1,\ldots\ldots,\ m \tag{3-5}$$

$$\sum_{i=1}^{m} \alpha_i\, y_i = 0 \tag{3-6}$$

Where:

- x = training elements
- y = class labels for x
- w = weight vector
- b = bias threshold
- C = correction factor
- α = Lagrange multiplier(s)
- m = number of elements

Algorithm 3-1 shows the basic SMO algorithm in the form of pseudo code.

Algorithm 3-1: Sequential SMO

```
1     input: set of training data $x_i$, corresponding labels $y_i$, $\forall \in \{ 1\ldots l\}$
2     output: weight vector $w$, $\alpha$ array and SV
3     initialize $\alpha_i \leftarrow 0$, $f_i \leftarrow -y_i$, $\forall\{ 1\ldots l\}$
4     compute $b_{high}$, $i_{high}$, $b_{low}$, $i_{low}$
5     update ${}^{\alpha}i_{high}$ and ${}^{\alpha}i_{low}$
6     repeat
7         update $f_i$ $\forall \in \{ 1\ldots l\}$
8         compute $b_{high}$, $i_{high}$, $b_{low}$, $i_{low}$
9         update ${}^{\alpha}i_{high}$ and ${}^{\alpha}i_{low}$
10    until <exit conditions>
11    update b threshold
12    store updated $\alpha_{i1}$ and $\alpha_{i2}$
13    update $w$
```

Where:

- x = training elements
- y = class labels for x
- w = weight vector
- b = bias threshold
- l = training data index set
- α = Lagrange multiplier(s)

In Algorithm 3-1, line 3 initializes the necessary structures, primarily the α multiplier and the objective function. Lines 4 – 9 show the core SMO optimization process. Every iteration is based on the selection and optimization of two Lagrange multipliers, subsequently the objective function. Line 10 checks for the exit criteria based on respective Karush-Kuhn-Tucker conditions (*KKT*), summarized in Equation 3-7. Line 11 updates the bias threshold.

$$\begin{aligned} \alpha_i = 0 &\Rightarrow y^{(i)}(w^T x^{(i)} + b) \geq 1 \\ \alpha_i = C &\Rightarrow y^{(i)}(w^T x^{(i)} + b) \leq 1 \\ 0 < \alpha_i < C &\Rightarrow y^{(i)}(w^T x^{(i)} + b) = 1 \end{aligned} \qquad \textbf{[3-7]}$$

In actual implementations, multiplier selection is commonly based on heuristics. Variations exist, which adopt different selection strategies. Lines 12 and 13 store the Lagrange multipliers and update the final weight vector.

A typical out-of-the-box SMO algorithm is inherently sequential. Iterative processes make use and update common, global internal structures, arrays and counters. The core computational effort is reflected in the processing required to come up with and perform the necessary identification of the respective Lagrangians [175]. This is undertaken to subsequently infer the required set of support vectors. This is performed alongside computing and updating optimality conditions and the objective function.

The training data set size is a highly influencing factor in terms of performance. In the context of spam, a typical representation of the respective feature set into corresponding vector form can create considerably large training corpora. These in turn need to be employed as input to the adopted algorithm. As already indicated, one of the more prevalent approaches for general distributed processing can be the partitioning of input data [152] [26] [159]. An important observation here is the consideration whether and to what degree to apply such distributed processing, i.e. for training, classification or both. The actual requirements differ considerably and separate avenues may well be required. In this work, the focus is on the former, which is the training perspective.

An additional aspect that mandates consideration is the evaluation of batch (*or offline*) versus online SVM based classification. Furthermore, the 'relaxed' SVM approach similar to [21] can also be taken into consideration. The same context should be considered in terms of any pre-processing stages which may include feature selection and vector characterization stages. Here, numerous options are available including TF-IDF and binary representations for example. However further research work is required to identify which should be the most appropriate in the proposed context. Opportunities for the consideration of automated inference can also be considered, such as discussed in numerous works including [176] [177].

Various MapReduce flavours and derivative implementations exist. These include Mars [28], Phoenix [178], Hadoop [179] and Google's [31]. Hadoop has become one of the most popular implementations due to its early market accessibility and its open source nature which allows users to have in-depth insight of its operation. Hadoop was established as the implementation of choice because of such characteristics - imperative to establish and ensure the required degree of understanding of the behaviour and performance characteristics of the distributed Support Vector Machine in such a construct. This choice is strengthened by Hadoop's high performance (*for example, during 2008, a 900+ Hadoop cluster sorted 1 Terabyte of data in approximately 200 seconds* [180][181]), user base and support.

Applying a split training dataset approach in a Hadoop MapReduce scenario, each participating node can compute the objective function using the chunked data, employing separate distributed SVMs. These generate localized support vector sets, via associated M/R maps. The subsequent reduce step(s) aggregate and compute the final targets. Beside the inherent performance increase potential given the high throughput possibilities of this approach, this can also help mitigate challenges associated with stale (*in the context of currency*) data. This can be achieved by re-training on a more frequent basis than potentially possible using more traditional approaches. This can keep the classifier increasingly aware of much wider and up-to-date intelligence.

3.4.2 Distributed SVM with MapReduce

A typical approach to employ distributed computing techniques for this operation is presented in [19]. Similar to the work presented in [18], the computation of the objective function (Equation 3-4) in this work is performed as a Map operation in the context of MapReduce. However, the parallelization of the bias computation in [18], is performed in such a way that each optimization iteration requires a Map operation as well as a Reduce operation. This approach introduces a substantial network communication overhead due to the large number of iterations to be performed. In the proposed approach herewith, a single MapReduce step is employed for each data fragment (*partition*) which reduces the overall computation and network communication overhead significantly.

From a M/R perspective and in the context of Hadoop, the data splitting strategy reflects the number of MapReduce tasks that will be employed, the underlying cluster capability in terms of node count as well as overall training file size. Each Map task (MAP_1…MAP_n) will process the associated data chunk ($Data_{Chunk(1)}$…$Data_{Chunk(n)}$) and generate a respective set of Support Vectors ($SV_{set(1)}$ … $SV_{set(n)}$). For optimal performance, the number of of data chunks should be a reflection of the number of task slots available on the Hadoop MapReduce cluster. In this particular scenario, these are then forwarded to a single Reducer ($REDUCE_1$) which will contribute the respectively aggregated Support Vector Set (SV), weight (w) and bias (b) elements of the global SVM to a final learned model. Given that the focus herewith is a linear SVM model [182], in our prototype, the aggregation of the weight (w) and bias (b) elements are performed using a sum and average strategy respectively.

In a linear SVM, an optimal hyperplane is the one with the maximum margin of separation between the two classes, where the margin is the sum of the distances from the hyper-plane to the closest data points of each of the two classes. SVMs have been shown to be adaptable to the Kearns statistical query model and associated summation form, and thus fit well into the MapReduce framework [158][183]. Adopting a sum/average strategy for establishing the global weight vector and bias is thus both computationally light from an algorithmic perspective and recognized to be accurate [158].

From a M/R perspective, this strategy also allows for aggregation via a single Reducer, decreasing M/R framework overhead. The final output will be used as the final classification model including the necessary information for the objective function to be able to classify unseen data. A high level pictorial representation of this approach is shown in Figure 3-4. Each Map can run on different nodes within the MapReduce cluster. The number of concurrent tasks each nodes support is configurable.

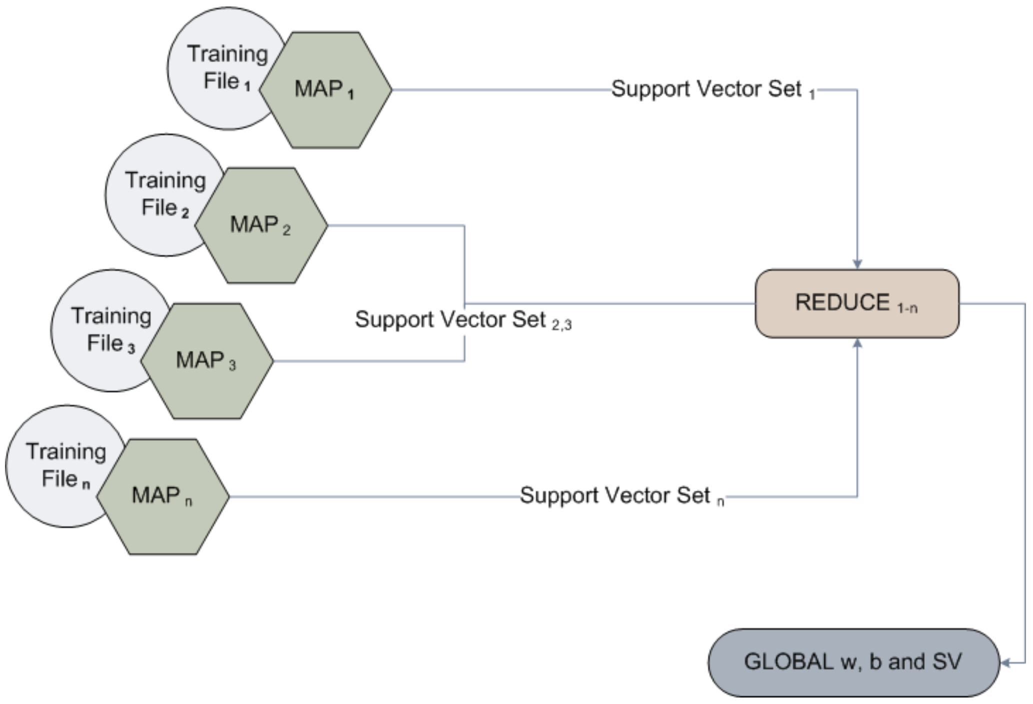


Figure 3-4: Aggregation of SVM process.

Algorithm 3-2 presents the distributed SMO based on the sequential one described in [11] and the outcome of similar work within the same research group [184] [185] [186]. The map segment of the algorithm is basically the same as the original SMO algorithm except that it is applied for each participating mapper. The primary difference lies in the ways that the global b threshold (b_{global}) and weight vector (w_{global}) are computed via the reducer, using an average and sum strategy respectively as described in the pseudo code.

Algorithm 3-2: MapReduce based distributed SMO

```
1    MAP_j ∀j ∈ {1…data_chunk}
```

```
2     input: set of training data $x_i$, corresponding labels $y_i$, $\forall \in \{1 \ldots l\}$
3     output: weight vector $w_i$, $\alpha_i$ array, $b_i$ and SV
4     initialize $\alpha_i \leftarrow 0, f_i \leftarrow -y_i, \forall \{1 \ldots l\}$
5     compute $b_{high}$, $i_{high}$, $b_{low}$, $i_{low}$
6     update ${}^{\alpha}i_{high}$ and ${}^{\alpha}i_{low}$
7     repeat
8         update $f_i$ $\forall \in \{1 \ldots l\}$
9         compute $b_{high}$, $i_{high}$, $b_{low}$, $i_{low}$
10        update ${}^{\alpha}i_{high}$ and ${}^{\alpha}i_{low}$
11    until <exit conditions>
12    update $b_i$ bias term threshold
13    store updated $\alpha_{i1}$ and $\alpha_{i2}$
14    update $w_i$
15    REDUCE
16    input: set of MAP_i weight vectors $w_i$, $\forall_j \in \{1 \ldots data_{chunk}\}$, set of MAP_i
      bias $b_i$, $\forall_j \in \{1 \ldots data_{chunk}\}$
17    output: global weight vector $w_i$, average b and SV
18    compute $b_{global} = \sum_{j=1}^{Map_i} b^{data_{chunk}} / Map_j$
19    compute $w_{global} = \sum_{j=1}^{Map_i} w^{data_{chunk}}$
```

where:

- Map_j = MapReduce Map
- $Data_{chunk}$ = training data associated with Map_j
- x = training elements, y = class labels for x
- w_j = local (Map_j) weight vector
- b_j = local (Map_j) b threshold
- l = training data index set
- α_j = Lagrange multiplier(s)
- b_{global} = global b threshold
- w_{global} = global weight vector

In Algorithm 3-2, for each $Data_{chunk}$ a Map (Map_j) operation is performed. In this context, line 4 initializes the necessary structures, primarily the α multipliers and the objective function. Lines 5 – 10 portray the SMO optimization process. Iterations are based on the selection and optimization of two Lagrange multipliers, subsequently the objective function. Line 11 checks for the respective exit conditions, whilst line 12 updates the bias threshold accordingly. For actual implementations, multiplier selection is frequently based on approaches such as heuristics, albeit strategies vary with specific implementations. Lines 13 and 14 store the Lagrange multipliers and update the local weight vector for the specific map (Map_j).

In contrast with the sequential SMO algorithm presented in [11], two additional steps using the reduce phase of the MapReduce prototype are performed. Basically, the reducer takes the following steps:

1. Performs an average computation on all respective b outputs emitted by the individual Map (Map_j) operations => b_{global} (Line 18)
2. Performs a sum operation on the weight vectors emitted by the respective Map (Map_j) operations => w_{global} (Line 19)
3. Performs the above two steps to generate the global b and weight vector for subsequent classification.

Weka's SMO [56] implementation as a baseline solver has been employed. For this part of the work the focus is on linear SVMs, although the approach can be easily extended and applied to non-linear variants as well. The base SMO algorithm is decomposed and re-structured to benefit from MapReduce. Each MapReduce map processes and associated data chunk in its entirety.

The output of each map process is the localized (*per data chunk*) SVM weight vector (Algorithm 3-2: w_j) and the bias (Algorithm 3-2: b_j) threshold. The primary role of the associated Reduce phase is to compute the global weight vector (Algorithm 3-2: w_{global}) by summing the individual maps weight vectors. The bias thresholds from each map output are averaged by the respective reduce phase (Algorithm 3-2: b_{global}). More formally, each individual Map_j is the partial weight vector and the value of b for the respective Datachunk partition (Equation 3-8):

$$\vec{w}_{partial} = \sum_{i=1}^{\ell} y_i a_i \vec{x}_i \qquad \textbf{[3-8]}$$

The global weight vector is computed via the Reducer by summing all partial weight vectors from each respective mapper (Equation 3-9) as well as averaging all b values for all Datachunk (Equation 3-10). The value of b is also handled by the reducer which averages the value of b across the partitions/mappers. To calculate the SVM output, the global weight vector and the value b are required (Equation 3-11):

$$\vec{w}_{global} = \sum_{i=1}^{n} \vec{w}_{partial} \quad [3\text{-}9]$$

$$\bar{b} = \frac{1}{n} \sum_{i=1}^{n} b_{i\,partial} \quad [3\text{-}10]$$

$$u = \vec{w} \cdot \vec{x} - b \quad [3\text{-}11]$$

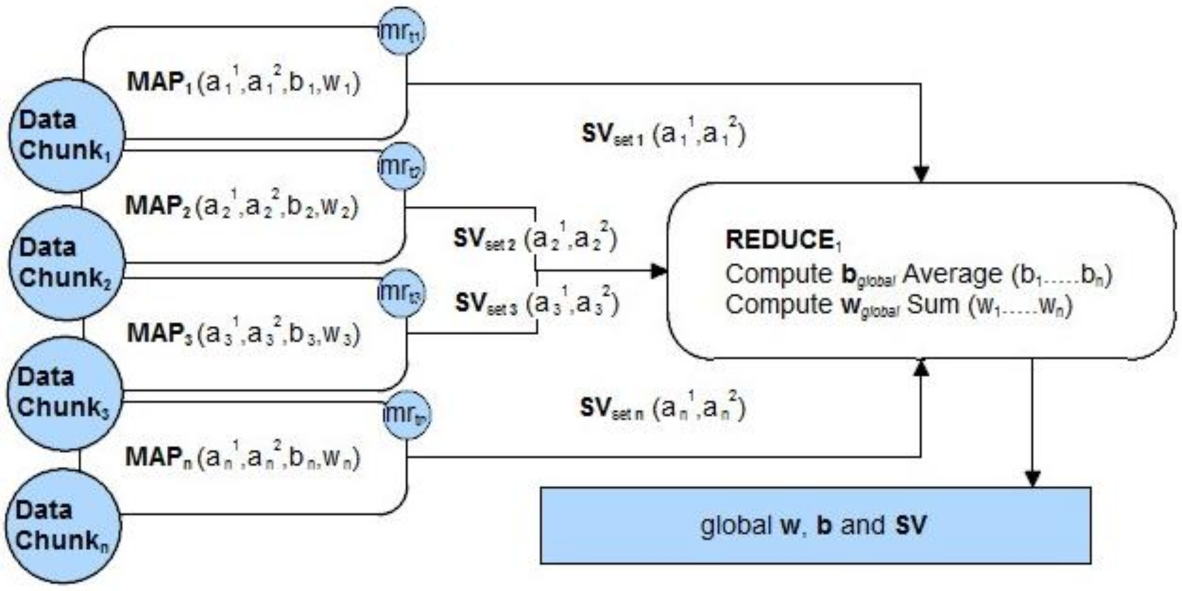


Figure 3-5: Reducer aggregation of individual, distributed SVM output.

For a non-linear SVM [187], each mapper's output would be the alpha array for the respective local partition and the value of b. The Reducer would join the partial alpha arrays to produce the global alpha array. As with the linear SVM approach, the reducer deals with the value of b averaging its value of across partitions.

In this respect, from a time complexity perspective, the original single sequential representation, i.e. O(m^2n) can now be contextualized in a MapReduce environment and expressed as (Equation 3-12):

$$O\left(\left(m^2 n / S\right) + n \log(S)\right) \qquad \textbf{[3-12]}$$

where:

- n = the dimension of the input
- m = the training samples
- S = the number of MapReduce nodes

Whilst as already stated SVM training is typically a global optimization problem and training an SVM by splitting the input data may reduce the overall accuracy [188], the empirical results identified and presented herewith further on demonstrate that real world accuracy degradation is minimal in this context using MRSMO. Furthermore, by virtue of the high performance infrastructure provided by a typical MapReduce framework, re-training the model with less concern to the timing and processing elements (*without ignoring them*), should this be necessary to increase accuracy, is much simpler. One can also perhaps consider adopting an overall strategy that targets training speed rather than accuracy as the primary focus.

To further (*or adopt a different approach*) mitigate to a certain degree the accuracy challenge, multiple MapReduce passes can also be considered and performed. In this context the output, which will have the final set of support vectors can be used as inputs for another MapReduce pass. Another avenue which can be explored is classifier combining, whereby different learning algorithms are applied during a phased approach. Output based on one type of training scheme can be employed subsequently as input to other different training methods. This approach is similar to that discussed in [189]. Other approaches including those based on incremental schemes but employ a singular approach can also be considered, as discussed in [190].

3.5 Experiments and Results

An appropriate spam dataset was required to be able to perform the required experiments. Table 3-1 presents a number of spam sources available for public scrutiny. The column designated type in Table 3-1 represents the corpus type – namely whether it is based on Text (T) or Images (I).

Table 3-1: Spam Corpora Examples

Corpus	**Location**	**Type**	**Year**
Enron-Spam	http://www.aueb.gr/users/ion/data/enron-spam/	T	2006
SpamBouncer	http://www.spambouncer.org/downloads/spamdata.shtml	T	2006
SpamBase	http://archive.ics.uci.edu/ml/datasets/Spambase	T	1999
ECUE	http://www.comp.dit.ie/sjdelany/dataset.htm	T	1999
Ling-Spam	http://www.aueb.gr/users/ion/data/lingspam_public.tar.gz	T	N/A
TREC 2006	http://plg.uwaterloo.ca/~gvcormac/treccorpus06/	T	2006
TREC 2007	http://plg.uwaterloo.ca/~gvcormac/treccorpus07/	T	2007
ImageSpam	http://www.cs.jhu.edu/~mdredze/datasets/image_spam/	I	2007
Princeton	http://www.cs.princeton.edu/cass/spam/	I	2007
CCERT	http://www.ccert.edu.cn/spam/sa/datasets.htm	T	2005
SpamAssassin	http://spamassassin.apache.org/publiccorpus/	T	2003

For most of the experiments carried out herewith, the SpamBase [191] dataset was employed. This is a multivariate dataset containing 4601 mail instances and formulated as a series of 58 attributes or feature representations (*independent variables, x_i in Equation 3-4*). These are mostly derived from character or word frequencies that describe the original mail message content. Around 40% of the dataset is spam.

The overall SpamBase structure is described below:

Table 3-2: SpamBase structure

48 real attributes which represent the percentage of words in the e-mail that match specific words
6 real attributes which represent the percentage of characters in the e-mail that match specific characters
1 real attribute which describes the average length of uninterrupted sequences of capital letters
1 integer attribute of type describing the length of longest uninterrupted sequence of capital letters
1 integer attribute reflecting the total number of capital letters in the e-mail
1 nominal {0,1} class attribute designating whether the e-mail was considered spam (1) or not (0)

A SpamBase instance takes the following basic form:

```
0,0.64,0.64,0,0……………………………………….0,0,0,0,0,0,0,0,0,0,0.778,0,0,3.756,61,278,1
```

The last identifier is the nominal class label which designates whether the email represents ham or spam – this is the dependent variable (*y in Equation 3-4*). In the experiments involving this data set carried out herewith, the entire feature set was employed during respective tests.

A number of experiments were carried out to identify the accuracy and efficiency of the distributed SMO when compared with the sequential counterpart. Weka's default SMO parameters were employed, namely c (*complexity*) set to 1.0, epsilon (*round-off error epsilon*) set to 1.0E-12, Polynomial Kernel. For file splits for which the training instances are less than the original number of instances in the entire dataset, Weka's resampling (*weka.filters.unsupervised.instance.Resample*) *without* replacement filter feature was employed, varying Weka's resample filter *samplePercentSize* parameter accordingly. For the rest, i.e., where the total number of instances is greater than what is available in the original training file, Weka's re-sampling *with* replacement was employed. The approach adopted to come up with testing instances (*unseen data, known but unsupplied label, for model accuracy validation*) was also based on the resampling approach. Approximately 200 instances were employed for the accuracy tests outlined - the same set was employed for both the sequential as well as the distributed SMO tests.

A baseline experiment to identify the sequential SMO performance using the SpamBase dataset on a typical desktop machine was performed. The desktop machine configuration is shown in Table 3-3.

Table 3-3: Desktop configuration

Hardware Environment	
Processors & Ram	Intel 2.33 GHz Dual Core, with 2GB Ram
Software Environment	
SVM	Weka 3.6.0 on Ubuntu 9.10

3.5.1 Performance of the Sequential SMO

The original dataset was split into a number of smaller sub-sets and extended to create larger input data sets where required, using the methodology described earlier. Weka's SMO classification scheme was employed [56], using a number of unlabelled instances and varying the number of training instances. Figure 3-6 shows the processing times of the sequential SMO algorithm during training.

3.5.2 Accuracy of Sequential SMO

From Figure 3-6, it can observed that the number of training instances varied from 204 to 128,000 with the training time ranging from 1 second to 563 seconds. Based on the 4601 instances of the SpamBase dataset, a varying training input size ranging from 204 and 128,000 training instances was employed. The sequential test failed when an attempt with 327,750 elements was performed. Respective accuracy ranged from a minimum of 82.04% correct to a maximum of 94.03% correct as shown in Figure 3-7.

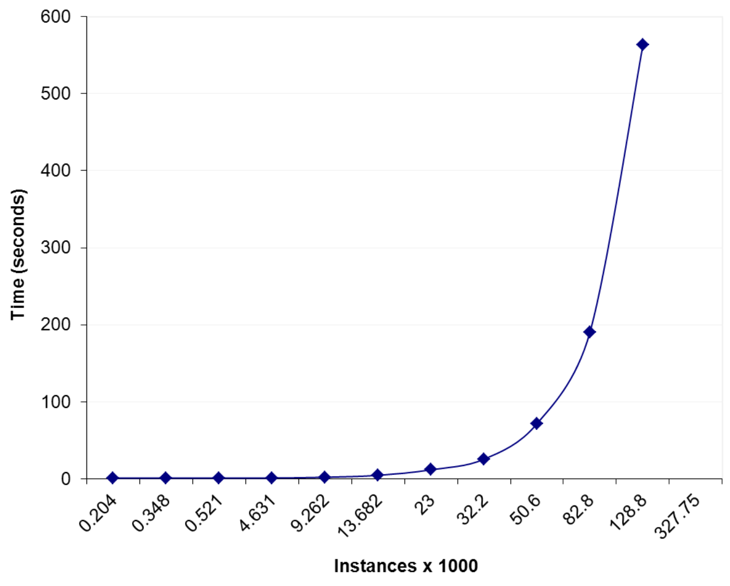


Figure 3-6: Sequential SMO training time

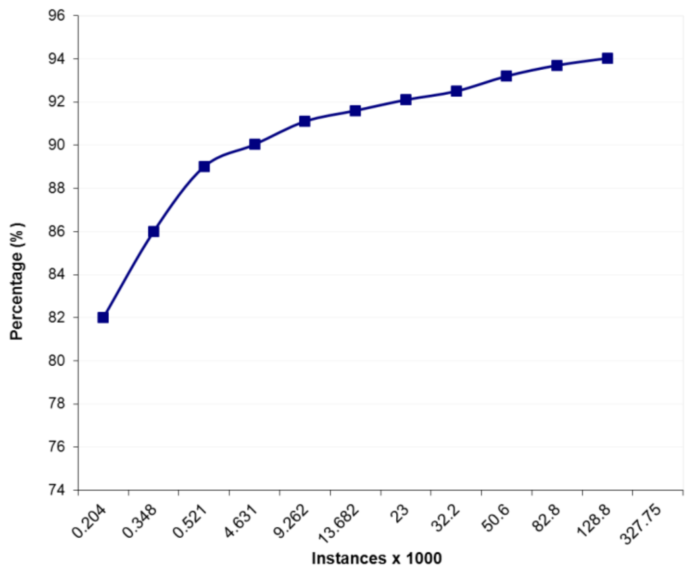


Figure 3-7: Accuracy of the Sequential SMO

3.5.3 Efficiency of Distributed SMO

In a second experiment, the approach taken for the sequential SMO was re-modelled for testing on a MapReduce Hadoop cluster. The SMO algorithm provided in Weka was extended, configured and packaged as a MapReduce job.

Each Map launches an instance of the Weka SMO (*SMO.java package*) on respective participating node. The actual MRSMO implementation is an extension of this implementation with the added functionality required by Hadoop MapReduce. This includes the ability to serialize, via Hadoop's MapReduce API's '*Writable*' interface, the individual distribute SMO models bias and weight vectors to HDFS. This is required to provide the Reducer with the opportunity to iteratively go through the individual distributed SMO's output and perform the respective individual model aggregation. This is done by reading the corresponding models output serialized by the Map operations. The Hadoop cluster was configured with the resources as shown in Table 3-4.

Table 3-4: Hadoop cluster configuration

Hardware Environment		
	CPU	*RAM*
Node 1 & 2	Intel Core Duo	2 GB
Node 3	Intel Quad Core	4 GB

Node 4	Virtual Machine on Node 3	512 MB
Software Environment		
SVM	Weka 3.6.0 (SMO)	
O/S, Hadoop and Java	Ubuntu 9.10, Hadoop 0.20 with JDK 1.6	

The time required to train the SMO sequentially using 128,000 instances on a single computer node was ≈ 563 seconds whilst the distributed SMO took ≈ 134 seconds using the experimental Hadoop cluster with 4 computer nodes. Figure 3-10 compares the efficiency of the sequential SMO in training with that of the distributed SMO using a varying number of nodes.

3.5.4 Hadoop MapReduce Speedup and Efficiency

Figure 3-8 portrays the speedup of the MapReduce approach compared with the sequential counterpart using an increasing data set size and number of participating nodes. The diagram shows that the MapReduce environment is more effective when the number of instances increases as well as the number of MapReduce nodes increases.

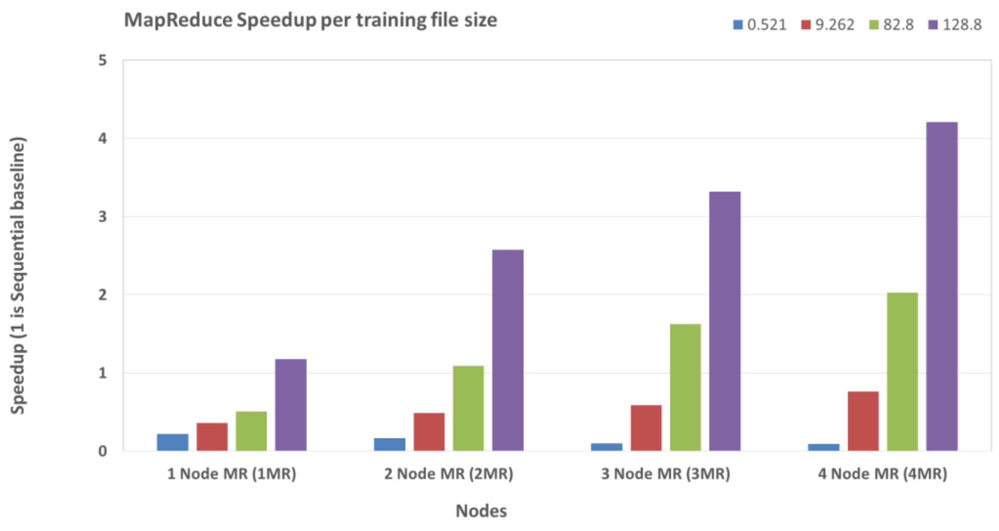


Figure 3-8: Hadoop MapReduce Speedup

Whilst speedup continues to improve with the increasing number of participating nodes, Figure 3-9 shows that with this particular experiment setup (nodes and number of training instances) efficiency is at best using a combination of 2 nodes and 128,000 instances.

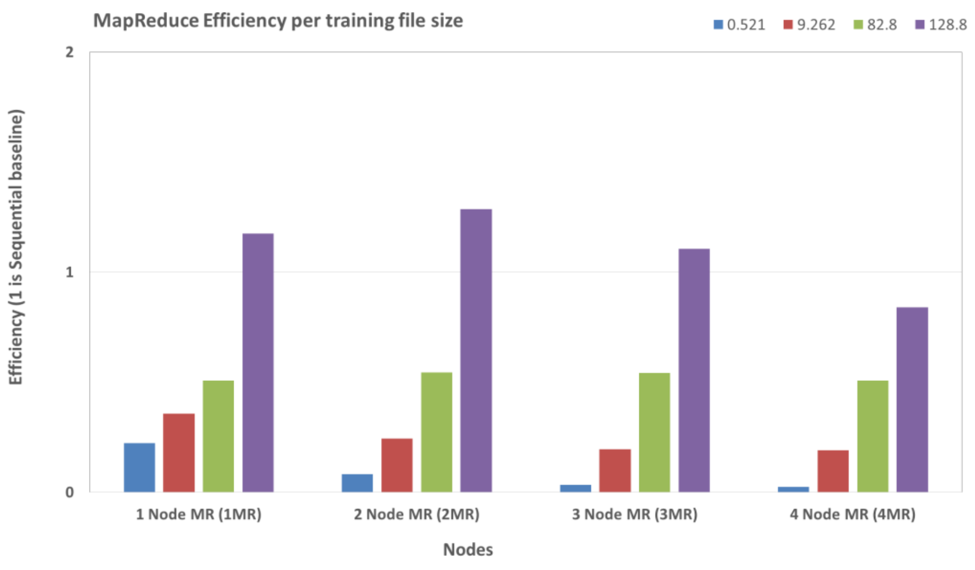


Figure 3-9: Hadoop MapReduce Efficiency

3.5.5 Accuracy of distributed SMO

Figure 3-11 shows that the accuracy of the distributed SMO using 4 MapReduce nodes is comparable to that of the sequential one. Using the global b and weight vectors obtained through a MapReduce classifier training run which employed ≈ 4600 instances, the average accuracy of the distributed SMO was ≈ 88% correct in classification. For the case of 327,750 instances, the accuracy was ≈ 92% correct in classification.

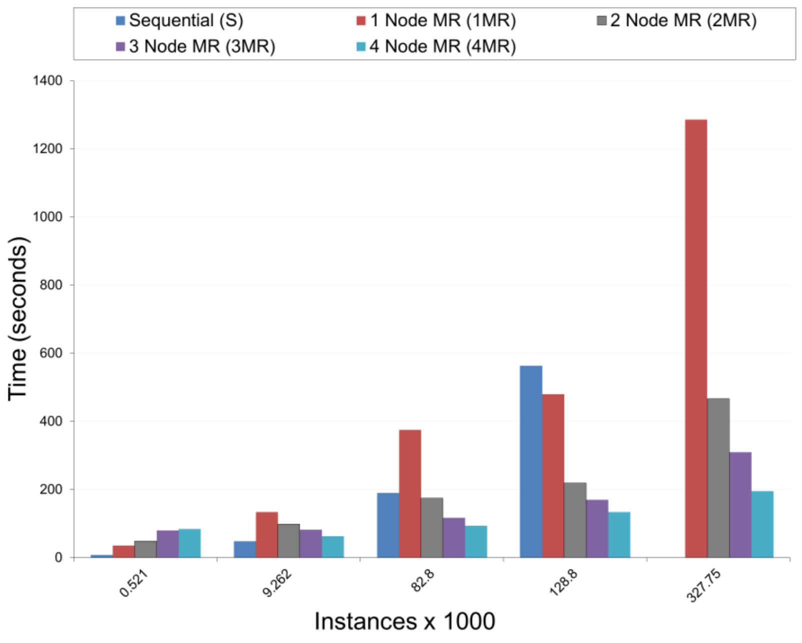


Figure 3-10: Performance of the distributed SMO

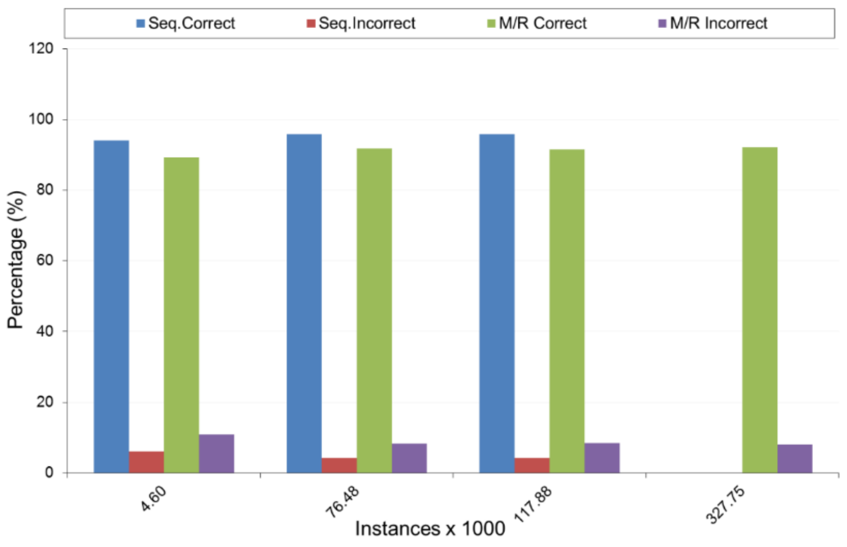


Figure 3-11: Accuracy of distributed SMO

Table 3-5 provides an overall performance comparison between the sequential SMO running on 1 computer and the distributed SMO running in a cluster of 4 MapReduce computers.

Table 3-5: The performance of the distributed SVM using 4 MapReduce nodes.

	Sequential	**MapReduce Average** (based on **8** *mappers*)
Correctly Classified	≈ 94.03 %	≈ 92.04 %
Incorrectly Classified	≈ 5.97 %	≈ 7.96 %
128,000 instances (training time in seconds)	≈ 563 s	≈ 134 s

3.5.6 Scalability of the distributed SMO

Given an appropriate number of processing nodes and map tasks, training the SVM using the proposed MapReduce approach evidently reduces training time considerably. This also provides increased scope for possible re-training. The accuracy of the distributed SMO is comparable to that of the sequential approach. Furthermore, Hadoop MapReduce has shown a near linear scalability for batch type jobs [192] [193].

Figure 3-12 shows the basic throughput of the distributed SMO in terms of the number of training elements processed per second with respect to the increase in number of processing nodes. The training dataset constituted of 327,750 elements. From the performance shown in Figure 3-12, it can be observed that the distributed SMO training also presents a near linear performance in spam filtering.

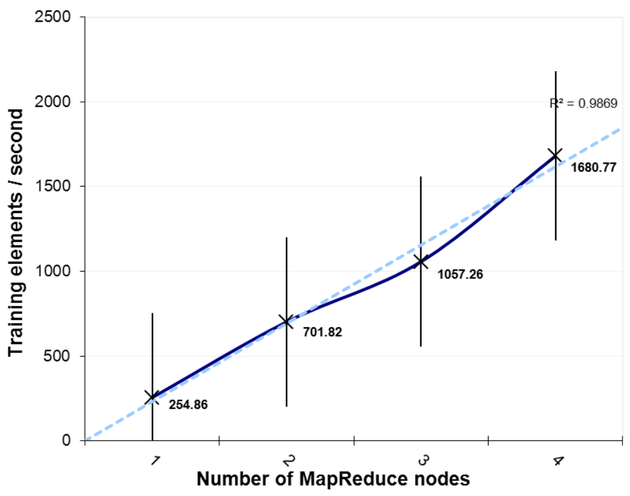


Figure 3-12: The scalability of the distributed SMO

3.5.7 HSIM

In order to evaluate further and confirm or otherwise this work in terms of scalability, HSim [194] was employed – a simulator designed and developed within the same research group carried out in this research work - to simulate the performance of the distributed MapReduce. In this work's context, it has been employed to evaluate the performance of the distributed SVM algorithm using a significant number of participating nodes.

The performance figures established during the experimental tests were used to establish a simulator configuration baseline. Subsequently, the underlying baseline parameters to replicate the same performance characteristics as close as possible, were tuned. Figure 3-13 shows the comparative performance characteristics of the experimental cluster and the HSim simulator.

The overall throughput and behaviour characteristic of the simulator are considered to be representative of the physical cluster counterpart [194].

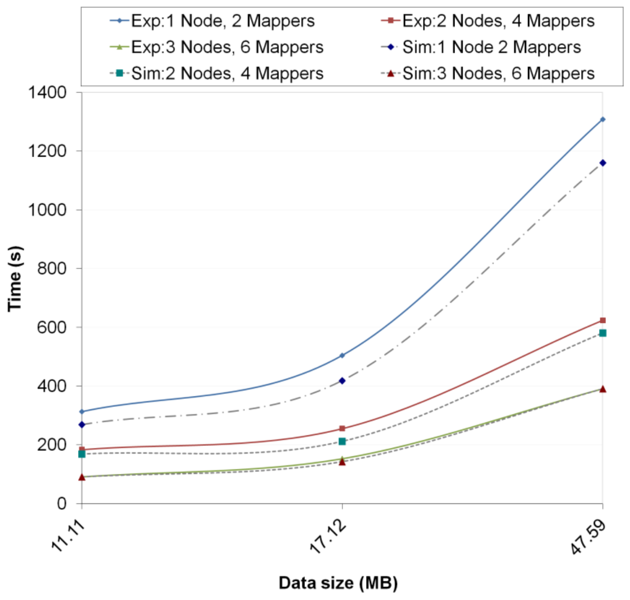


Figure 3-13: Experimental and Simulator Performance Baseline Indicators

Based on this configuration, a series of HSim based simulation runs and experiments were carried out to study the potential behaviour of the SMO MapReduce algorithm using different simulated cluster configurations. Key observations and outcomes of these simulation runs are presented in the following set of figures, namely from Figure 3-14 to Figure 3-17.

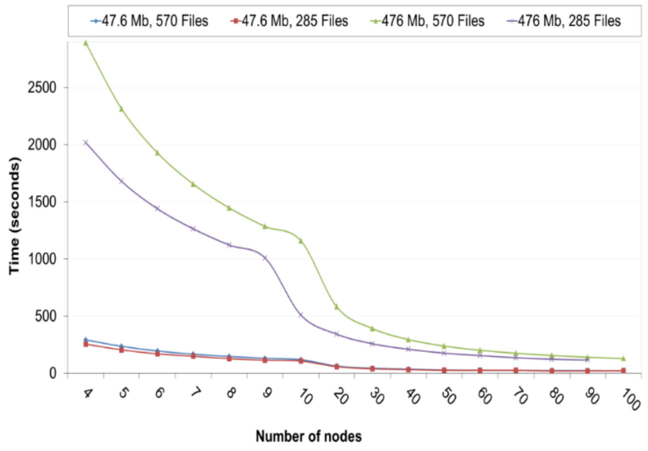


Figure 3-14: Varying the number of input files and size

Figure 3-14 shows the time (*in seconds*) employed to process 47.5 and 475 MB of training input data respectively, split across 570 and 285 input file sets and applying a varying number of processing nodes, namely between 4 and 100. Each node employed 2 mappers.

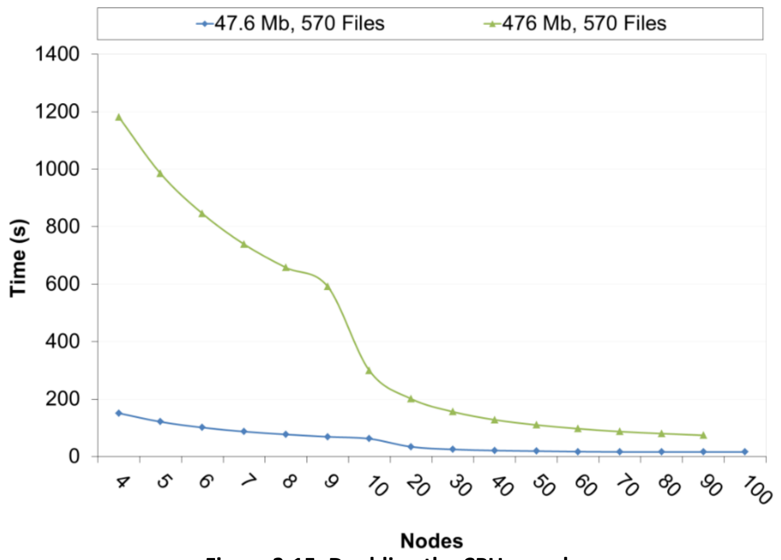


Figure 3-15: Doubling the CPU speed

Figure 3-15 illustrates the time (*in seconds*) to process 47.5 and 475 Mb of training input data respectively, split across 570 input file sets as in the experiment presented in Figure 3-14, but doubling the simulated processor performance.

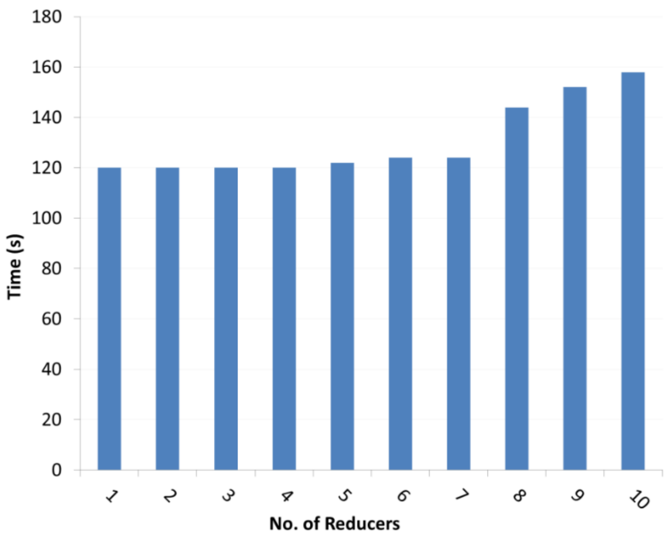


Figure 3-16: Increasing the number of reducers

Figure 3-16 presents the simulated performance characteristics of the algorithm using an increasing number of reducers. Given the approach adopted in the actual implementation, namely, there is only one single Reducer required (*explicitly set by MRSMO*) to perform the individual distributed SVM output aggregation, having more than one Reducer is ineffective. The simulation experiment shows that increasing the number of Reducers will actually result in a degradation of performance – this is due to associated management overhead from a

Hadoop M/R perspective. Figure 3-17 shows the aggregated simulated performance increase of the proposed distributed SMO algorithm as the number of nodes increases.

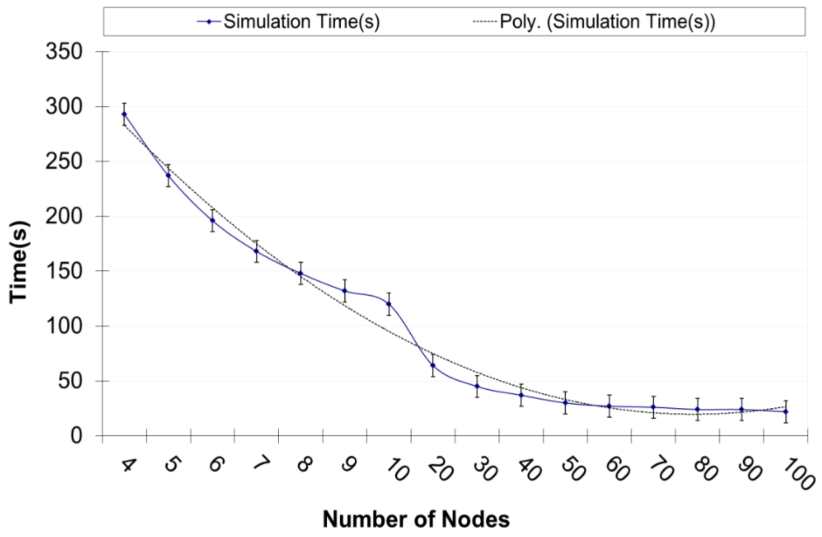


Figure 3-17: Hadoop M/R SVM simulated scalability

The primary observations from the simulation runs demonstrate that increasing the number of participating nodes significantly reduces the processing time correspondingly. The same applies to increasing raw CPU performance. It has also been observed that the greater the training input size (*file/s*), in conjunction with appropriate number of mappers, achieves better throughput when compared to smaller yet more numerous input file sets. A key influencer here is the reduced disk I/O to CPU utilization ratio - maximizing CPU load is evidently better than employing additional disk interaction. On the other hand, the introduction of additional reducers does not bring any substantial benefits in the applied context. This is primarily due to the relatively 'simple' compute operations performed by our algorithm (*namely sum, average – Algorithm 2*) at this (*Reducer*) stage.

3.6 Comparison with MPI

To further assess the performance of the MapReduce based SVM, MRSMO is compared with the MPI based parallel SMO algorithm presented in [18]. The maximum speedup recorded via the MPI approach is 21 times using a 32 processor configuration. The MapReduce based SVM was evaluated using the same Adult dataset [195] adopted in the MPI work. This specific data set, containing 48,000 instances, with 14 attributes (*independent variables*), predicts whether income exceeds $50,000 per year based on census data (*class label, dependent variable*). Two sets of tests, namely using a standard Polynomial kernel and a subsequent Gaussian kernel were carried out. The proposition of the experiments was to compare the performance of the parallel distributed approaches. Following the definitions presented in [18], speedup and efficiency in the context of MapReduce can be defined as follows:

Speedup = Sequential SMO time / Parallel SMO time **[3-13]**

Efficiency = Speedup / Number of processor cores **[3-14]**

The configuration of the MapReduce Hadoop cluster employed for this particular set of experiments is shown in Table 3-6. Hadoop's default cluster configuration and scheduling is employed. A single TaskTracker is employed on each node. The micro-cluster for this experiment set was made up of 3 physical nodes with a total of 8 CPU cores and 3 virtual nodes each making use of one of the physical cores as summarized below.

Table 3-6: Hadoop configuration.

Hardware Environment			
	CPU	*Cores*	*RAM*
Node 1 & 2	Intel Core Duo	4	2 GB
Node 3	Intel Quad Core	4	4 GB
Node 4, 5 & 6	Virtual Machine on Node 1, 2 & respectively	(Virtual)	512 MB
Software Environment			
SVM	WEKA 3.6.0 (SMO)		
OS, Hadoop and Java	Ubuntu 10.04 - Hadoop 0.20.2 - JDK 1.6		

3.6.1 Speedup – Polynomial and Gaussian Kernels

Figure 3-18 and Figure 3-19 present the respective speedup and efficiency results, in seconds and percentage improvements, over the baseline distributed SMO performance of the MapReduce SVM using the 2 kernels.

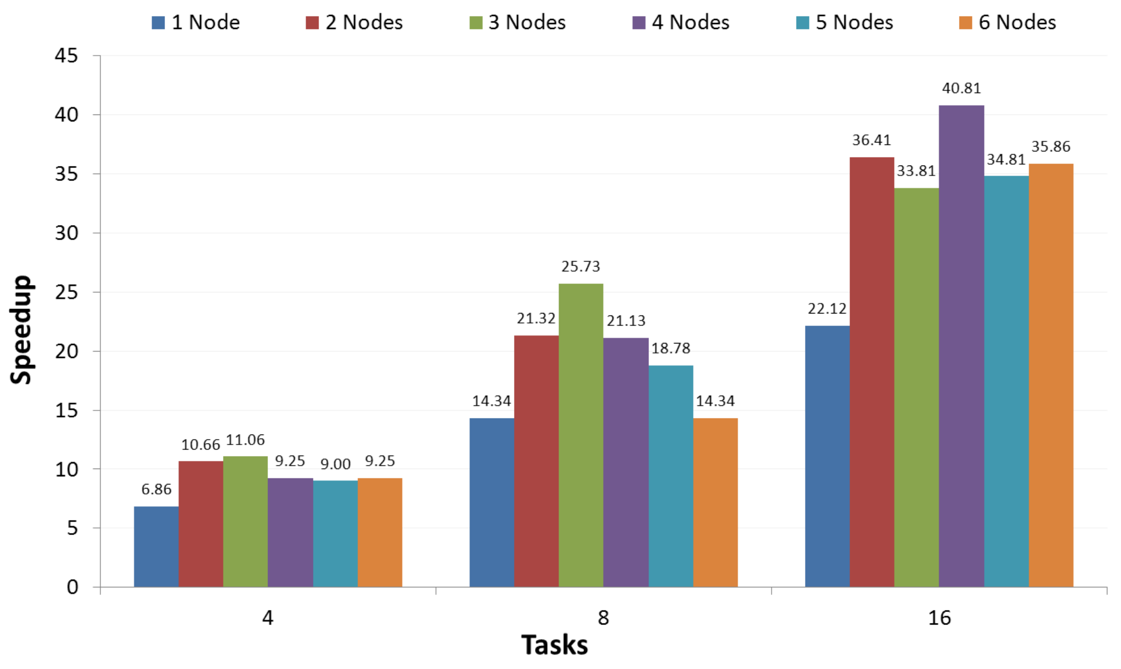


Figure 3-18: MRSMO - speedup - Polynomial Kernel

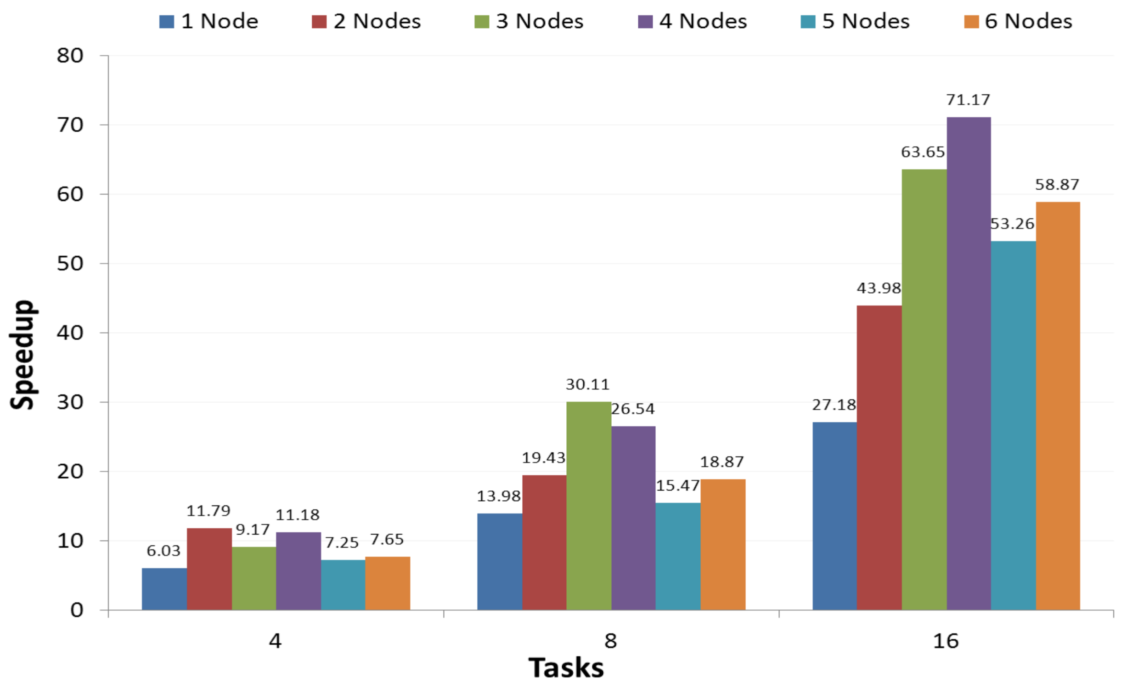


Figure 3-19: MRSMO - speedup - Gaussian Kernel

3.6.2 Efficiency – Polynomial and Gaussian Kernels

Figure 3-20 and Figure 3-21 present the efficiency of the MapReduce SVM using the two kernels respectively.

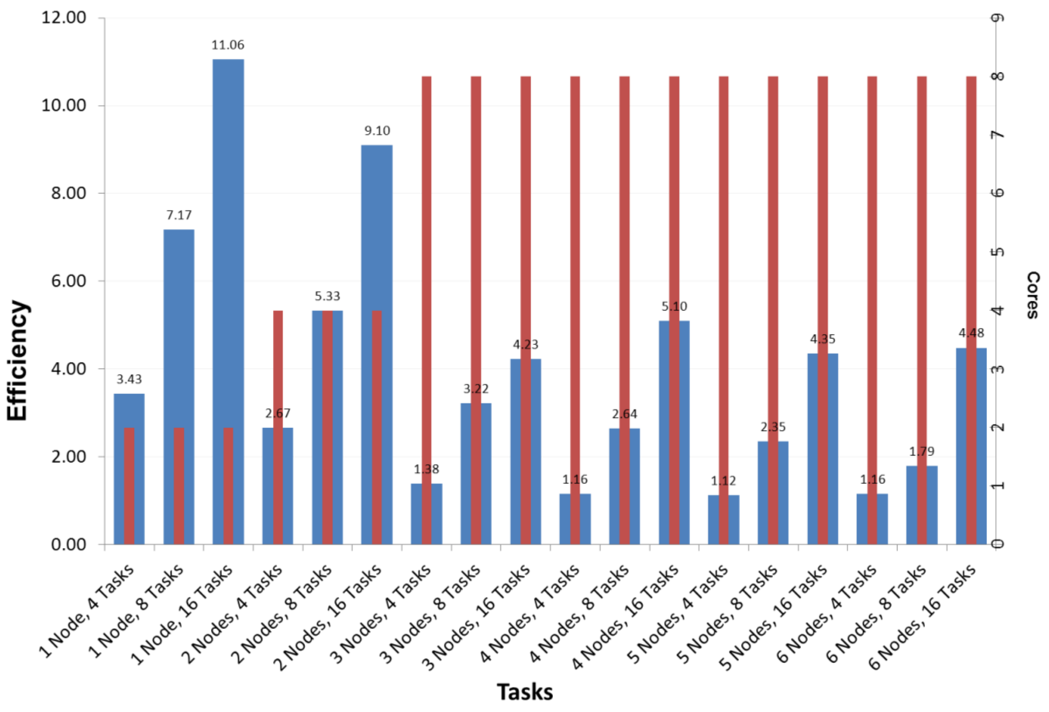


Figure 3-20: MRSMO - efficiency - Polynomial Kernel

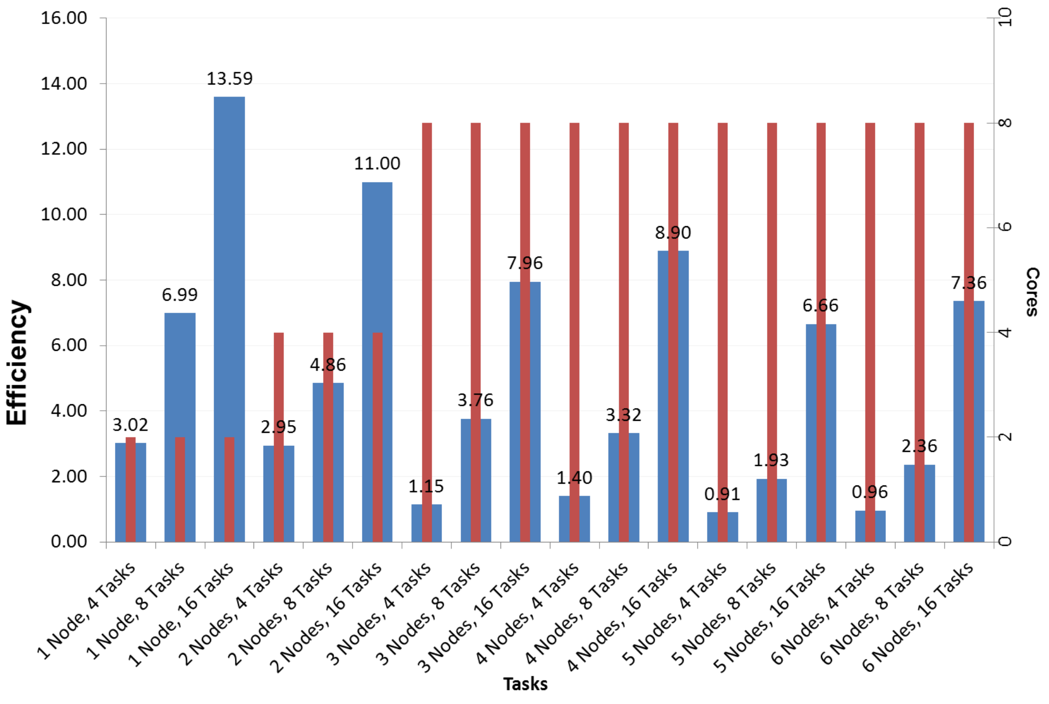


Figure 3-21: MRSMO - efficiency - Gaussian Kernel

3.6.3 MPI results discussion outcome

For the MPI speedup and efficiency experiments, the number of MapReduce nodes were varied from 1 to 6 in conjunction with the number of input files, which were adjusted between 4, 8 and 16 (X axis), for a total of 48,000 instances. Each individual file split is processed by a MapReduce map task. Therefore, for 4 splits, a total of 4 files containing 48,000/4 training instances each using 4 map tasks were processed. For 16 splits, a total of 16 files containing 48,000/16 training instances each using 16 map tasks were processed. The 'red' bars represent the number of CPU cores that the particular configuration was executed on. The average speedup (*in seconds*) and efficiency (*in percentage*) improvement of the MapReduce based SMO algorithm using the Polynomial kernel were about 20 times and 4 times respectively compared with the sequential SMO. The corresponding results from the MapReduce SMO algorithm using Gaussian kernel were about 28 times and 5 times.

From these results it can be concluded that the MapReduce based SMO is more efficient than the MPI based approach which has an efficiency of 21/32. The primary reason for this is that the MapReduce based SMO fully distributes the dataset onto a number of computing nodes reducing the overhead in training significantly. In the MPI based approach, only the update computation to the f_{array}, b_{up}, b_{low}, i_{up} and i_{low} are performed in parallel. The rest of the SMO algorithm is performed sequentially on a single CPU. Furthermore, the MPI approach also has to deal with the overhead associated with retrieving and converging the global b_{up}, b_{low}, i_{up} and i_{low} respectively.

3.7 Summary

This chapter presented an overview of the typical SVM algorithm, focusing on the Sequential Minimal Optimisation implementation. The decomposition of the typical global SMO optimization approach is re-constructed in a MapReduce construct with the objective to parallelize the algorithm. This is performed to improve the scalability as well as performance of the machine learning training perspective. The performance and accuracy of the parallel and sequential SMO is shown, illustrating that with 8 mappers and 128,000 training instances there is a 4x improvement. The accuracy, whilst reasonable, suffers a bit – this is due to the data split approach for training the SVM solver.

On the other hand, this research shows that with the MapReduce based approach, the scalability perspective improves quasi-linearly. The HSIM simulator is employed to provide further insight with respect to establishing the impact of varying the underlying M/R computing capabilities, identifying the effect of increased data file sizes and number of nodes. A comparison and evaluation with a typical MPI based approach is also discussed in terms of speedup and efficiency, using the Gaussian and Polynomial Kernels for non-linear SVM training.

The chapter is finalized by a discussion of work related to the application of SVM techniques from a general perspective as well as that which is more focused on spam filtering.

CHAPTER 4 – RDF Assisted Distributed SVM

As already cited, SVM training is a global optimization problem which typically relies on the dataset in its entirety to infer the final objective function. As also confirmed by the respective MRSMO experiments in Chapter 3, training an SVM by splitting the input data set and working on the individual sub-sets separately, reduces the overall accuracy [188]. Alongside performance, accuracy of spam filters is however key. Possibilities of how the degradation of accuracy introduced by distributed SVM computation can be improved and how to possibly seek and capitalize on end user contribution to the same extent are explored in this chapter. Traditional ensemble approaches as well as an ontology based feedback loop pattern are also briefly discussed and evaluated.

The chapter is organized as follows. Section 4.1 presents a discussion on related work. Section 4.2 provides an introduction and exploratory evaluation of accuracy improvement using traditional ensemble approaches. Sections 4.3, 4.4 and 4.5 provide the context for and subsequently present the ontology based feedback loop for the MRSMO's accuracy improvement. Section 4.6 evaluates this improvement. Finally, section 4.7 summarises the key elements of this chapter.

4.1 Related Work

The incorporation of domain knowledge facilitates, improves automated filtering processes as well as increases the scope for classification accuracy in the context of spam filter training and classification. By virtue of the inherent increased 'readability' and 'expressiveness' elements, ontologies can provide end users with wider opportunities in terms of better understanding and subsequently contribution towards improving spam filtering.

In this work, SPONTO acts as an RDF based enabling feedback loop base for the training and classification processes. Whilst the approach is similar, this work contrasts with that presented in [104] where the ontology itself is employed for classification. Furthermore, in this work the focus is on the high throughput SVM classifier training scheme rather than the ontology. The latter is employed specifically for accuracy improvement. This enables the proposed approach to scale - velocity and volume are key characteristics of the modern spam challenge and which are not considered nor tackled in the construct presented in [104]. In the work presented [103], ontology semantics are on the other hand employed to reflect more appropriately user preferences. Putting greater emphasis on the personalization aspect increases the perceived usability greatly. In this respect, the authors propose an adaptive ontology for email filtering using a J48 decision tree based approach based on a pre-trained WEKA model. This model is subsequently translated into an RDF based ontology representation, similar to this work. Jena is also employed for the generation of the actual ontology from the WEKA decision tree model – basically, the RDF representation of the WEKA decision tree is used as Jena input for ontology generation. The ontology representation generated by Jena subsequently provides a number of assertions which are subsequently employed to classify email as spam or ham. This is in contrast with the work proposed in this dissertation whereby the ontology dimension is specifically employed for intelligence augmentation. In this proposed work, the actual spam filter training is performed using the proposed, distributed SVM - believed and shown to be highly scalable and offering performance characteristics that cannot be easily achieved using traditional sequential training schemes.

In their work designated "*On Enhancing the Performance of Spam Mail Filtering System Using Semantic Enrichment*" [196], the authors expose the common cold start issues related to traditional classification such as Bayesian. The authors highlight the problems associated with emails where there are few key terms that can be employed for the analysis and subsequent classification processes. The work proposed subsequently presents an approach towards the generation of concepts from emails. In this approach, concepts are characterized by typical term occurrence or frequency. A set of sub-concepts or candidate concepts are computed and associated with each concept. The measure of candidate concept (*sub-concepts*) similarity is performed by calculating respective term vector cosine similarity. This provides the opportunity to improve the classification of email with few terms by enriching key concepts with associated sub-concepts. Specifically with respect to the classification dimension, the proposed work focuses on Bayes classification and its challenge related to limited number of terms. Unfortunately, as presented, there is no discussion of practical, real-world performance evaluation nor comparison with other classification schemes and approaches in terms of scalability and overall effectiveness.

A slightly different approach and associated set of considerations are discussed in [197]. Here, the authors present a discussion on two key different perspectives for spam filtering enhancement or enrichment. The first consideration is performed at infrastructure level. The second perspective looks at the application dimension. First, multiple filters based on ensemble learning [198] are considered and adopted. In this work, filtering is implemented using a derivative of the two-phase ensemble learning algorithm. From an application perspective, an approach referred to as operable email is studied. Operable email in the presented context refers to a number of key research dimensions intended for the application of intelligent, agent based applications. These can perform sophisticated knowledge based tasks in an autonomous fashion on behalf of their users such as the one presented, Email Centric Intelligent Personal Assistant (ECIPA). There is however little detail of how the operable email actually works. It is also understood that the recommended approach requires considerable changes to the overall email ecosystem to be able to achieve its effectiveness – something which poses a number of challenges in terms of acceptance and adoption from the wider email service provider and user community.

In [102], the authors consider and discuss the challenge related to the personal element of spam as well – an email which is spam to someone may be considered and treated as perfectly legitimate to someone else. As discussed earlier, this amplifies the need towards ensuring the necessary degree of user control in terms of preference. A combination of Bayesian and ontology approaches are considered for email classification in this work. Once again, ontology space is used for user preference formalization by introducing concepts including white list, categories and keywords. A category per user association is introduced. This feature provides the ability to identify and correlate the type of content and associated categories that the recipient normally expects from a particular sender (*or set of senders*). The initial relationship between the keywords that relate to specific categories is based on weights. These weights are used in identifying the probability that a typical keyword belong to a particular category. Subsequently, Bayesian classification is applied for actually classifying the mail by computing the probability of the email content belonging to a particular category. This biggest advantage of this approach is the inherent simplicity to integrate with current approaches. The biggest challenge remains however where different personalized filters exist for a large user base. Scalability becomes a challenging prospect where the complexity and size of individual personalized filtering increases significantly.

Concept extraction and identification is a challenging process, both manually and even more so automatically. A very interesting approach for automated identification and extraction of concepts is described by Yang and Callan in [199]. This is achieved by means of a number of techniques including nominal n-gram text mining from input corpora. Exact and near duplicate candidate concepts identification (*based on tokenization*), part-of-speech tagging as well as clustering techniques are also employed. Basically, ontological concepts are created via the identification of hypernym relationships using WordNet. Ontology hierarchy is created using a modified K-mediods clustering algorithm. Concept naming is tackled rather interestingly as well. Google search is employed in this respect – a set of current concepts are submitted to the search engine as a single comma delimited string. Term frequency identification is applied on the top 10 returned results snippets and the most common word in this respect is selected as a new concept. The results described provide an interesting insight on the accuracy that can be achieved in this respect. Taghva et al [200] tackle the identification of features using a different, ontology based approach – this not in the specific construct of spam filtering however. This approach presented is however, in part, similar to the approach researched in this dissertation work – a specifically built ontology forms the basis of the presented CLIPS based inference approach. The CLIPS instances represent emails. Together with the respective class ontology definitions generated, instances are employed for feature set identification. In turn these are employed as a training set base for a Bayesian classification scheme. Whilst this work is not contextualised in a spam filtering construct as already indicated, scalability in a spam filter training and filtering scenarios are fundamental. As proposed, the CLIPS and Bayesian based approach are not believed to be able to perform to the same levels of performance and scalability of the work presented in this dissertation if they were considered for such application.

Concept drift is an acknowledged challenge in model learning tasks [201]. The work presented by Han et al [202] focuses on concept drift and adaptive learning from a spam filtering perspective. This reflects the standpoint that users may change their opinion with respect to interest and subsequently classification of certain mail types. Similar in part to [102], this work considers and employs end user actions and interaction characteristics to augment intelligence and identify user preference for spam classification. To further increase accuracy, the authors also study and analyse end user's intent with respect to actions undertaken. Founded on statistical correlation between terms within text corpora and intended to mitigate lexical matching challenges as well as concept drift, latent semantic indexing is basically a modified Support Vector Machine Kernel. Single Value Decomposition is applied to the entire source corpora to estimate respective term usage, alongside additional steps to control various non-representative influencing factors. This approach is applied by [203] in the work titled "*Using Latent Semantic Indexing to Filter Spam*". The author compares precision and recall using traditional approaches and one which employs LSI, with an interesting outcome especially with respect to spam classification recall. The basic challenge here remains, as indicated by the work itself, the semantic base which is subject to reconstruction when additional source content needs to be introduced. Similar to [103] and [196], in [204], the authors propose the use of semantic alongside syntactic techniques towards spam filtering. The objective is to reduce user involvement for spam filtering updating. By integrating user interest and representing it as ontology, the approach links content type based on interest and which is expected to be received from the recipient. Scores are attributed to emails which relate to whether there is a relationship between the sender and expected content and interest.

A noticeable increase in the number of spam messages that employ images to deliver their message has become apparent. The rise of image based spam further increased the interest in considering alternative approaches to how this new mutation can be possibly controlled, including those presented in [162] [163]. Image spam inflicts even greater problems in terms of processing power and bandwidth requirements – subsequently incurring even greater cost. The work presented in "*Using Visual and Semantic features for anti-spam filtering*" [205] takes this consideration from an ontology perspective. The key techniques applied include latent dirchlet analysis [206], latent semantic analysis and indexing [207] as well as optical character recognition. Hsia and Chen describe another approach for image based spam detection [105]. In this work, a scheme based on exploiting hidden topics within images is employed. These 'latent' topics are identified and subsequently employed as training input for a binary classifier. The authors describe a probabilistic approach to infer hidden semantic meanings that are represented as images. In [99], the image element of spam representation is tackled using a traditional approach, based on optical character recognition. Term frequency–inverse document frequency (*TF-IDF*) feature set selection is applied. Additional processing is performed for converting the model generated via the machine learning scheme adopted using Weka to RDF (*Resource Description Framework*), similar in part to this work. This step is employed to be able to generate the required ontologies, subsequently employed to create custom user filters.

4.2 Ensemble Approaches

Various techniques can be applied in a machine learning construct to improve classifier accuracy. Ensemble schemes provide one such opportunity in this respect [64] [65]. Bagging and boosting approaches for example are statistically known to improve accuracy in general. However, the degree of actual accuracy improvement (*or degradation rate*) largely depends on the context [66]. Context influencers include classification algorithms, parameterization as well as dataset properties.

To explore initial possibilities in this regard, an extension to the prototype was applied and a simple experiment to identify any immediate improvements using the proposed M/R approach and the SpamBase dataset was performed. Any overall accuracy improvement would have to be considered in the context of any increased computational complexity introduced by the respective processes.

Two sets of tests were performed. One based on Weka's bagging (*sampling with replacement*) method and the other on SMOTE (*Synthetic Minority Over Sampling Technique*). Bagging [208] involves the random generation of training sets and combining subsequent classifications using the same base classifier. SMOTE [209] focuses on the over-sampling of both minority and majority class. For these tests, 8 input splits (*files*) were employed using the entire 4600 base instances of the SpamBase data set.

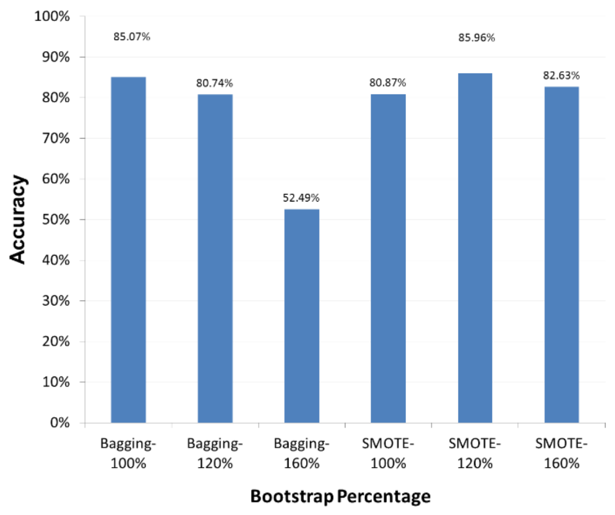


Figure 4-1: The effect of Bagging and SMOTE

Figure 4-1 portrays the outcome of the basic evaluation of the application of SMOTE and Bagging. In this exploratory experiment, the distributed MapReduce SMO classifier was seeded with a 100%, 120% and 160% training set over-sample spread across the 8 input splits to evaluate the influence on accuracy using both techniques. The results indicate that in this particular context the recorded accuracy diminished slightly.

Using bootstrap aggregation, the distributed SMO classifier achieved a maximum accuracy of 85.7%, reduced to 52.45% when the sampling with replacement target was set to 160%. The corresponding figures for the SMOTE based approach where 85.96% and 80.87% respectively. Accuracy degradation could be partly influenced by the number of instances evaluated, data and ensemble sizes. These factors have the potential to negatively impact SVM learning when using ad-hoc ensemble techniques. Accuracy in this context is also influenced by the current bias and weight aggregation approach in the respective Reducer phase of MapReduce.

4.3 Ontology

The formal modelling, specification and representation of real world elements as a set of inter-linked concepts within a domain describe basic ontology. Each individual concept represented and any inter-relationships within the ontology are fundamentally unambiguous. The semantics of ontology are commonly organized in a hierarchical fashion. The ontology hierarchy employs a relationship basis whereby represented concepts in the ontology domain are associated using an “*IS-A*” relationship between each other. Beside real world objects, ontology hierarchy also reflects and represents events and properties that describe the real world model. Ontologies also describe a number of facts or axioms. They also provide varying degrees of automated 'reasoning'. Reasoners and validators are an important facet to ontology development and application. The intelligence of inferring additional logical outcomes and facts is achieved with reasoners. Different types of reasoners exist, the more popular being those based on description logic – OWL DL for example is based on *SHIQ* Description Logic [210]. Others are based on probabilistic variants. Popular implementations of description logic reasoners include F-OWL, FACT, Jena/Jena2, OLWP, Euler, Pellet, Kaon2, OWLIM and SESAME. A number of comparison studies looking at the various characteristics and performance of reasoners have also been carried out [211]. Beyond reasoners, semantic validation, hence validators, then deal with the consistency, logic and quality control perspectives and requirements of ontologies respectively. Key semantic validation efforts are described in [212].

4.4 Assisting MRSMO's accuracy with RDF

The application of ontology and semantics in the context of spam filtering can assist in the definition and understanding of spam in a better and more formal way. The ability to exchange intelligence and subsequently the potential for machines to process it in a formal and interoperable fashion provides numerous opportunities. Annotating email messages with metadata also brings numerous benefits including supplemented intelligence, context richness and formalization. This increases the scope for collaborative spam filtering information exchange noticeably.

As indicated in chapter 2, the majority of ontology based work identified during this research and in this particular research context and scope, tends to be inclined towards the application of ontology based techniques for the description and representation of user preferences. This is believed to be primarily due to the simplicity of the approach as well as the contributed effectiveness towards improving spam filtering from an end user perspective.

The key challenge with user dependent or focused ontology creation is the complexity of the process itself. The ability to understand ontology creation and equally importantly providing simple intuitive tools for the end user to be able to actually develop them is not a trivial matter. The same applies to the expressiveness of the ontologies. OWL is not always considered expressive enough for some applications for example [213]. The respective learning curve required from an end user perspective should not be underestimated. "*Reasoning is hard*" [213] as well, as are complexities associated with scale. Where automated ontology formation is applied, beside the actual complexity of the algorithms involved, one must also keep in mind the scale of the computing requirements which may be required. This is in terms of actually creating preliminary ontologies and perhaps more importantly to keep them up to date. Vocabulary sizes and ensuring the necessary degree of focus are also important facets that need to be kept in consideration.

To date, there does not seem to be a standard ontology for email and spam representation. Furthermore, separate initiatives have a tendency to end up in the development of distinctive ontologies. This creates challenging situations with respect to ontological interoperability requirements. Generally speaking, ontology based interoperability is not trivial by any measure [106]. In the context of spam this becomes further amplified given the subjectivity aspects when applied in a global sense.

Spam filtering effectiveness can be increased via a combination of end user focus and contribution, as well as the consideration for large scale classification of email. Spam and spammers techniques evolve continuously over time to circumnavigate filtering. It thus becomes critical to ensure that a corresponding effort towards evolving and tuning of spam filtering approaches is applied. One way to achieve this is via a combination of high performance spam filter training algorithms as well as non-intrusive, simple RDF based intelligence augmentation schemes which allow the users to contribute and influence the spam filtering process outcome quality accordingly.

4.5 RDF Based Feedback Loop

Spam filtering approaches in terms of the actual decision making process varies. One can however note a number of typical, multi-level approaches in this respect. These include mail source, header and content. The first is not directly related to the email itself but rather to the propagation sources. Header and content analysis form the basis of most intelligence applied during classification.

A number of key perspectives are scrutinized and respective logic applied on them. These include, header formatting, source blacklisting, mail routing, header keywords, spam keywords and / or phrases in content or body, image content analysis, malformed URL's, text to HTML to image ratios and a number of others. An initial prototype ontology to describe the email domain in the context of spam classification, focusing on the SpamBase dataset as a baseline, has been explored. In order to improve the overall classification accuracy of the distributed SMO algorithm presented earlier, it is extended with an RDF based, feedback loop process.

As presented in [214], designated SPONTO (short for SPamONTOlogy), the prototype ontology is based on 3 primary concepts, namely Email, Ham and Spam respectively. Jena and Pellet for ontology manipulation and verification are employed. Evaluated via the Pellet reasoner [215], the current basic prototype ontology structure exhibits the following core proprieties described in Table 4-1:

Table 4-1: SPONTO - core properties.

Property	**Value**
Owl Profile	OWL 2.0 EL
DL Expressivity	AL(D)
Axioms	193
Logical Axioms	128
Classes	3
Data Properties	62

Where:

- OWL Profile defines the structure restrictions of the ontology
- DL Expressivity refers to the ontology complexity in terms of reasoning, conciseness and ease of understanding
- Axioms refer to logical statements that are related to the ontology roles and concepts
- Classes describes the number of classes within the ontology
- Data properties refers to the number of data properties in the ontology

The feedback loop is employed to re-train the distributed SMO with end user contributed intelligence. This is performed to mitigate the accuracy degradation challenge introduced with the training data file splitting strategy and respective separate SMO computation. SPONTO thus reflects all the basic elements presented in the SpamBase [191] dataset as well as additional attribute assertions.

SPONTO is also employed to provide users with additional information in terms of mail class (*Ham - Figure 4-2, or Spam-Figure 4-3*), the classification result of the distributed SMO as well as support for instance weights.

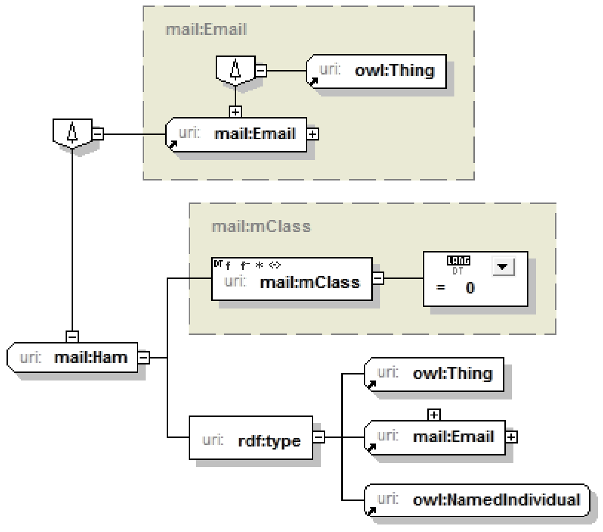


Figure 4-2: SPONTO ham instance

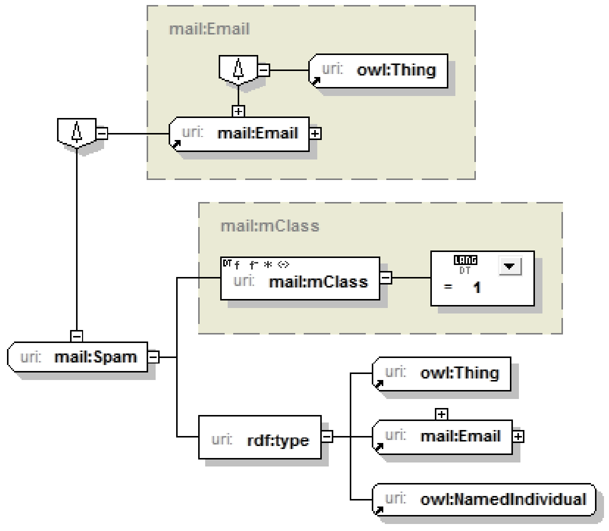


Figure 4-3: SPONTO spam instance

Intelligence and quality improvements conveyed through the supplementary instance attributes via end user contribution is employed for correcting and influencing training data. Beyond the baseline structure (refer to Table 3-2, Pg. 60), the key, additional attributes exposed via the underlying RDF schema are:

1. isMisclassified – whether the instance was classified correctly using the distributed SVM classification scheme. This is required to be able to provide the end user with the visibility of whether the instance currently being evaluated was classified correctly or otherwise using the distributed SVM.
2. instanceWeight – the relative importance the distributed SVM should give in training the classifier. This allows the end user to increase or decrease the relative importance of the specific instance.

The end user contributed intelligence supplemented training sets are subsequently employed for the regeneration of the classifier by the distributed SVM - Figure 4-4.

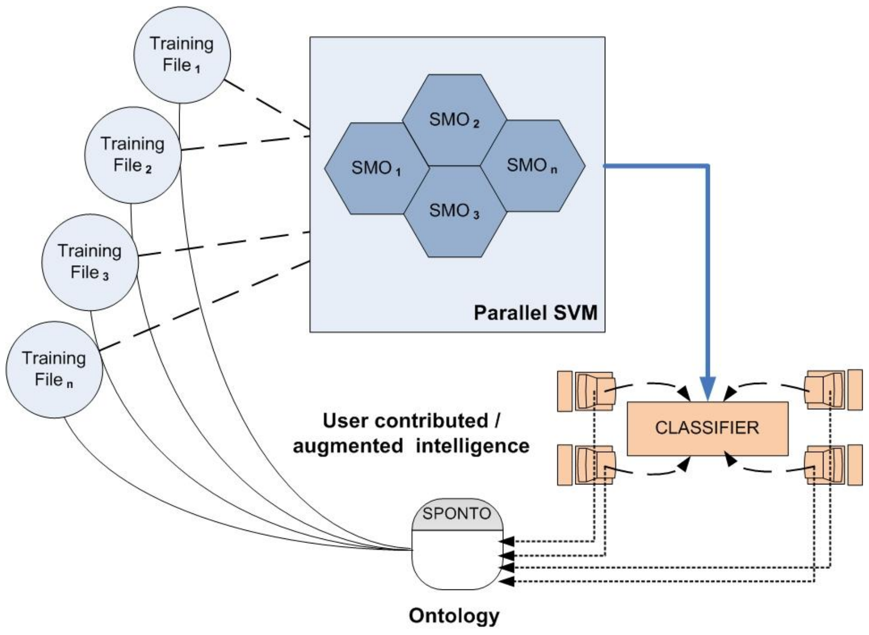


Figure 4-4: RDF assisted classification process

The key steps of the RDF feedback loop to improve the distributed SVM accuracy are as follows:

- Employ split data training sets with the parallel SVM to compute and output the SVM classifier.

- Use the distributed SVM classifier to classify Ham/Spam testing set and create respective SPONTO instances, reflected as an RDF based graph.
- Misclassified nodes are identified as a set of separate RDF instances.
- Misclassified instances from the RDF based representation are automatically correlated, through SPARQL, with the original training set data.
- Using a base RDF representation of SPONTO, end user contributes feedback/intelligence and corrects relevant instances.
- Merge user modified instances as well as corrected classifications.
- Re-generate the training data fragment(s) for the subsequent distributed SVM re-training.

This is represented below using pseudo code in Algorithm 4-1.

Algorithm 4-1: RDF based enhancement of training set

```
input: set of training data $x_i$, corresponding labels $y_i$ $\forall_i \in \{ 1\ldots l\}$
output: set of RDF based intelligence supplemented training data $y_i$
1.   compute SMO Model $SMO_i$ with via  MRSMO
2.   classify test data $y_i$ using $SMO_i$
                 generate base graph $B_G$ = {V,E} from $x_i$
                 $\forall_i \in \{ 1\ldots x_i\}$ {
                       $B_G$ triple=createTriple(subject$_i$, predicate$_i$, object$_i$)
                       $B_G$.addTriple(triple)
                       }
                 create QueryGraph $Q_G$ = {V,E} on $B_G$
                 execute SparQL query $Q_R$ on $Q_G$ where $B_G$.misclassified eq. true
                 create OAGraph $OA_G$ = {V,E} with:
                       $\forall$ $Q_R$ {
                             $OA_G$ triple=createTriple(oa$_{subjecti}$, oa$_{predicatei}$,
                             oa$_{objecti}$)
                             rdfAugmentedGraph.addTriple(triple)
                             }
3.   correct/personalize instances.
                 Correct / Increase instance weights if/where appropriate on $OA_G$
                 using Protégé
                 Update $B_G$
4.   merge modified instances as new training set
                 merge $OA_G$ with $B_G$
5.               Generate RDF Assisted OntS$_i$ from $OA_G$ via $OA_G \cup B_G$
6.   convert     to $y_i$
     OntS$_i$
```

The learned model output from the distributed SVM is applied on the instances which require classification. For each testing instance, a new RDF instance based on the SPONTO construct is generated.

RDF generation can take two different paths:

- The first is via the extraction of instance data and generation of respective, automatic SPARQL query generation and correlation. This query is applied on the base RDF graph to identify respective misclassified elements, *or optionally*
- By applying an intermediary J48 classifier. In this approach, the fact rules generated by the C4.5 decision tree algorithm are transformed into respective SPARQL queries. This approach provides a degree of classification outcome 'cross checking' – between the distributed SMO learned model and the J48 classifier. Once again however, the respective queries are executed on the base RDF graph to identify misclassified elements.

In either approach, misclassified nodes are identified as a set of final RDF instances. The misclassified instances from the RDF based representation are automatically correlated, through SPARQL [216], with the original training instances and the latter presented to the end-user.

Users can subsequently contribute feedback, preference and intelligence by increasing individual training instance weights, removing instances or modifying instance classification outcomes etc. For this work, the actual contribution and influencing step is performed manually, with the intent however that for future work this will be changed as indicated in Section 6.

From a high level perspective, the following example portrays the transformation of an original instance to one which the end user modified and contributed back to the training set, by changing the overall weight of the instance and changing the class label - (Ham/Spam) in this particular interaction. The interactions can obviously be applied to any other element(s).

Original instance:

Element	**Class**	**Weight**
{ SPAMBASE DATA }	0	1

User modified instance:

Element	**Class**	**Weight**
{ SPAMBASE DATA }	1	1.5

Where:

1. SpamBase data refers to the instance information (such as occurrence of a specific terms etc. Individual variables can be modified.
2. Class reflects whether this is Ham or Spam – within the instance records themselves this is represented as 1 or 0, however through the RDF to ARFF conversion scheme, the end user actually sees the more representative terms Ham or Spam.

In this example, weight reflects the relevance, based on end user perceived importance of the instance – the default weight is 1. For this example, this means that the end user augmented the importance (*weight*) of this particular instance.

From an implementation standpoint, a base RDF graph from the SpamBase ARFF file is generated based on the SPONTO construct. The transformation is performed using the WEKA [101] and JENA [217] API's respectively. A software tool has been implemented which integrates the respective WEKA and JENA functionality to transform ARFF instances to respective RDF triples by iterating over the ARFF training data (see Algorithm 4-1), using SPONTO as a baseline ontology, and represented pictorially in simple example presented in Table 4-2. This table shows an ARFF file fragment, with the initial header (line A1), followed by the respective SpamBase attributes (e.g. shown in line 2 - Table 4-2).

The instance type {Spam or Ham} is represented by the Class attribute (1 or 0) in line A4. The actual instances (line 6 onwards) then follow, prefixed by the data section (line A5).

Table 4-2: Weka to RDF conversion

	WEKA
A1	@relation spambase
A2	@attribute word_freq_make numeric
A3	. . . .
A4	@attribute class {1,0}
A5	@data
A6	0.4,......................{ SpamBase data }..,1
A7	. . . .
	RDF Representation
B1	<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:owl="http://www.w3.org/2000/01/rdf-schema#"
B2	<rdf:Description rdf:about="http://localhost/sponto/mail#oMail-3745"
B3	<mail:word_freq_make>0.4</mail:word_freq_make>
B4	..
B5	<mail:IinstanceWeight >1</mail:instanceWeight>
B6	<rdf:Mail rdf:class=" http://localhost/sponto/mail /spam"
B7	</rdf>

Further to this example transformation from ARFF to RDF, the attribute "*word_freq_make*" in a typical instance (Table 4-2, Line A2) is transformed as a corresponding RDF node with the subject being "*word_freq_make*" (Table 4-2, Line B3), the predicate "*isSubclassOf*" and object "Mail". Correspondingly, the entire instance will be transformed to the RDF triplet equivalent (Table 4-2, Line B3/B6), generalized in Table 4-2 and Table 4-3:

Table 4-3: RDF representation of ARFF instance

Subject	Predicate	Object
ArffInstance	isOfClass	Spam

For the feedback loop, Protégé, the RDF editor and knowledge acquisition system [218] is employed - for interaction and contribution. Any industry standard OWL/RDF manipulation software can be used however. Ontology reasoners can also be employed to explore and validate the generated RDF base - available as respective Protégé Plug-ins in this particular case and which include HERMIT, PELLET and FACT amongst others. This provides the ability to perform inference and assertion operations, identify inconsistent concepts and equivalency on the ontology. Additional interaction with the ontology, including extensive querying such as via Manchester Syntax [59] based DL-Query languages can also be performed to fine tune instance quality.

Protégé allows third party extensions to be developed and incorporated with the standard product via an application programming interface. For the prototype, a number of SPARQL template RDF operations for end user application are provided (see Figure 4-5). These template operations are deployed as a Protégé plugin. A number of examples are described in Table 4-4. The end user can therefore employ the provided template operations on the RDF to establish opportunity for instance quality improvement without in depth understanding of the underlying RDF operations mechanics. The opportunity to manually interact with the ontology is obviously still possible. Template operations can be added or removed accordingly.

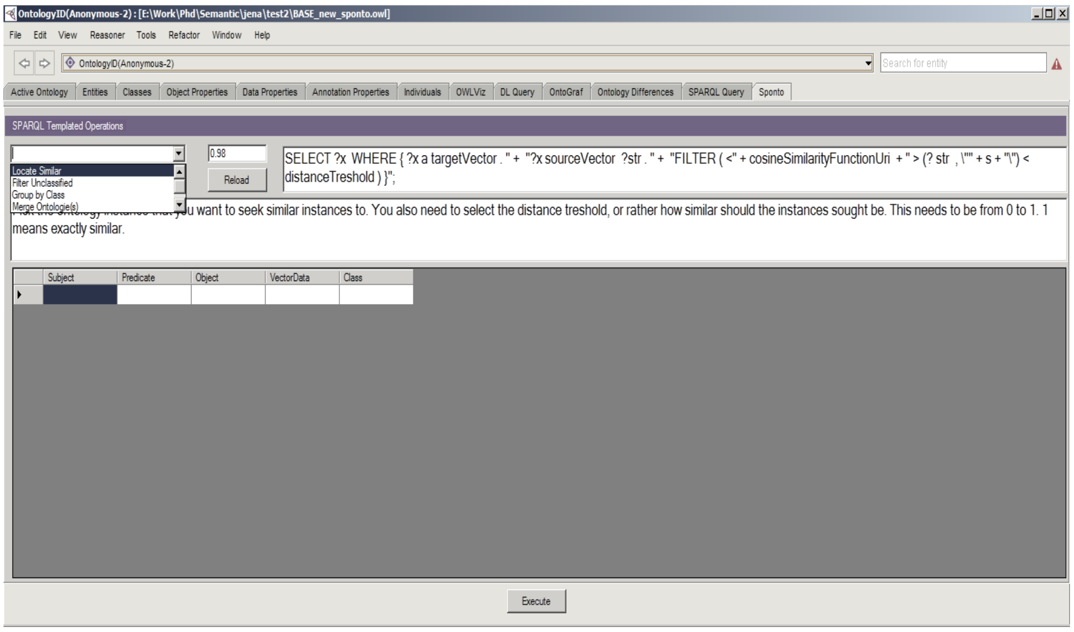
Table 4-4: Sample SPONTO Protégé Template operations

Operation	SPARQL Query Example
Locate similar RDF instance(s) using cosine similarity filtering, using a parameterized distance threshold.	**SELECT** ?x **WHERE** { ?x a targetVector . " + "?x sourceVector ?str . " + "**FILTER** (<" + **cosineSimilarityFunctionUri** + " > (? str , \"" + s + "\") < *distanceTreshold*) }"; *where* **cosineSimilarityFunctionUri** is defined in the form of $\boldsymbol{Instance}_{similarity} = \lVert \boldsymbol{targetVector} \rVert \, \lVert \boldsymbol{sourceVector} \rVert \cos \Theta$

Filter unclassified RDF instances	**SELECT** * WHERE { ?Email mail:mClass ?mClass .
	FILTER (xsd:double(?mClass) != 0 || xsd:double(?mClass) != 1) }

Group instances by class type	**SELECT** * **GROUP BY** ?mClass

The final step(s) of the process involves the re-generation of the Weka ARFF input files from the final RDF dataset for subsequent processing by the distributed SVM filter training method. The same approach with respect to the conversion from ARFF to RDF is employed, in reverse, from an implementation perspective – also using JENA and WEKA API's as baseline tools. Corrected instances are accordingly weighted and merged with the original input source. Training instance weighting for a degree of noise mitigation is also considered a simple yet effective way to further assist the accuracy improvement effort and provides a number of advantages when compared to discarding [219]. The overall classification process is portrayed in Figure 4-7. Figure 4-5 and Figure 4-6 portray the templated SPARQL operations and Protégé based interface.


OntologyID(Anonymous-2) : [E:\Work\Phd\Semantic\jena\test2\BASE_new_sponto.owl]
File Edit View Reasoner Tools Refactor Window Help
OntologyID(Anonymous-2)
Search for entity
Active Ontology Entities Classes Object Properties Data Properties Annotation Properties Individuals OWLViz DL Query OntoGraf Ontology Differences SPARQL Query Sponto
SPARQL Templated Operations
Locate Similar
Filter Unclassified
Group by Class
Merge Ontologie(s)
0.98
Reload
Subject Predicate Object VectorData Class
Execute

```
SELECT ?x  WHERE { ?x a targetVector . " +  "?x sourceVector  ?str . " +  "FILTER ( <" + cosineSimilarityFunctionUri  + " > (? str  , \"" + s + "\") <
distanceTreshold ) }";
```

…u want to seek similar instances to. You also need to select the distance treshold, or rather how similar should the instances sought be. This needs to be from 0 to 1. 1 means exactly similar.

Figure 4-5: SPONTO prototype template interface

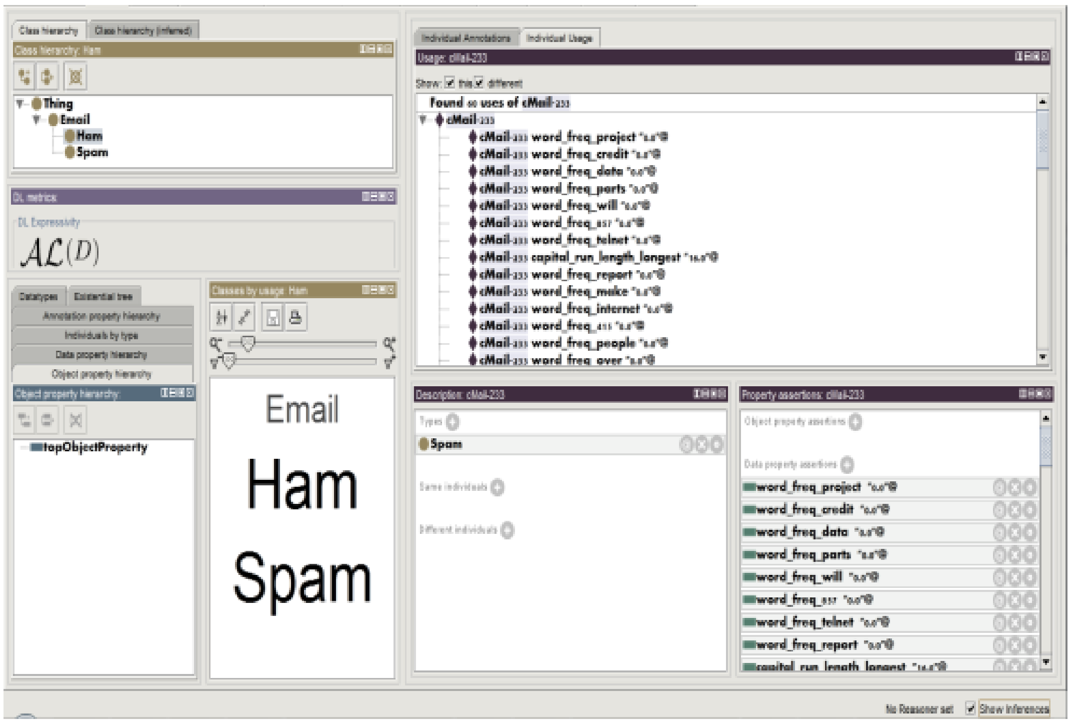


Figure 4-6: RDF based user assistance and enhancement via Protégé

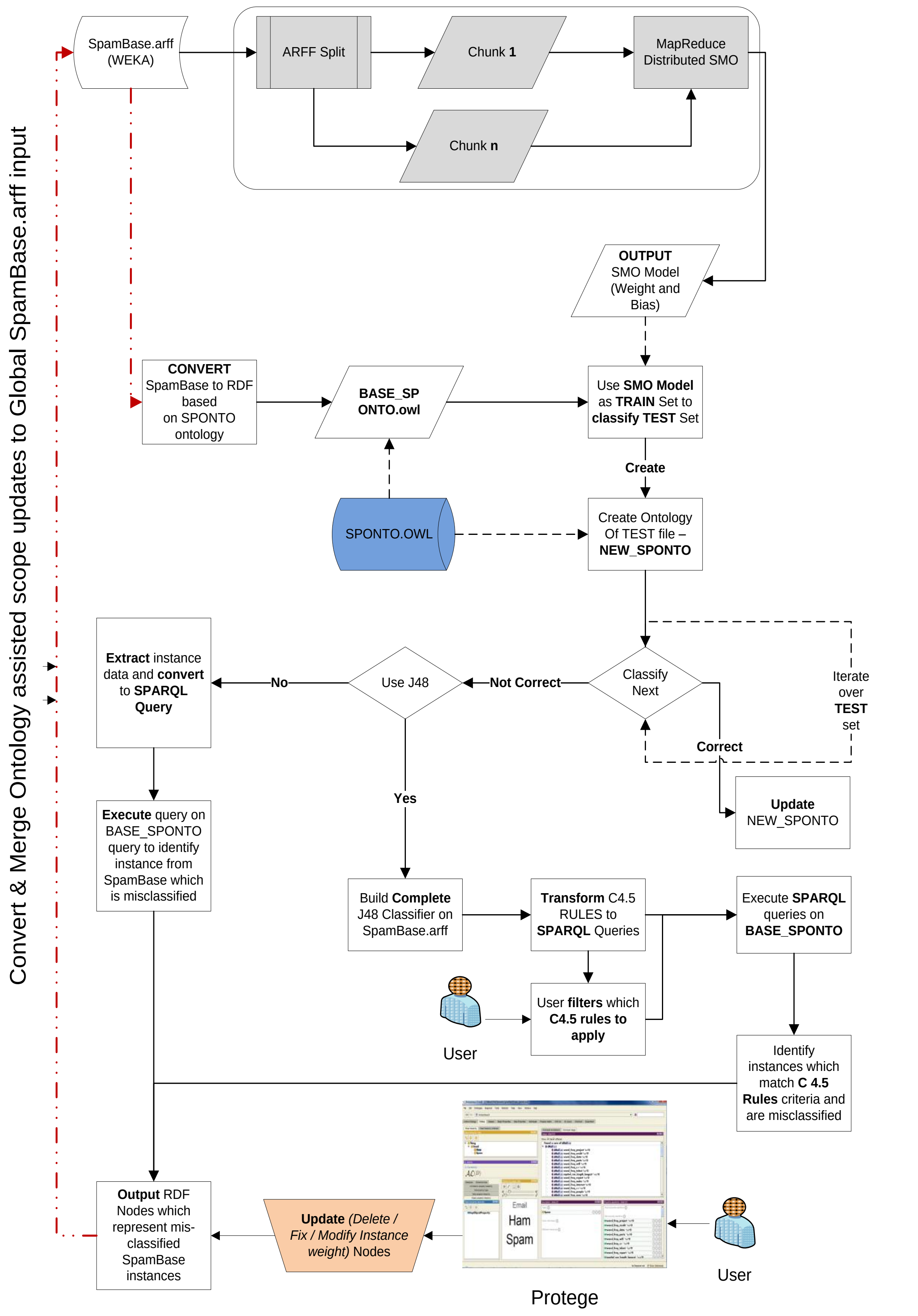

Convert & Merge Ontology assisted scope updates to Global SpamBase.arff input
SpamBase.arff (WEKA)
ARFF Split
Chunk 1
Chunk n
MapReduce Distributed SMO
OUTPUT SMO Model (Weight and Bias)
CONVERT SpamBase to RDF based on SPONTO ontology
BASE_SPONTO.owl
Use SMO Model as TRAIN Set to classify TEST Set
Create
SPONTO.OWL
Create Ontology Of TEST file – NEW_SPONTO
Extract instance data and convert to SPARQL Query
No
Use J48
Not Correct
Classify Next
Iterate over TEST set
Correct
Update NEW_SPONTO
Yes
Execute query on BASE_SPONTO query to identify instance from SpamBase which is misclassified
Build Complete J48 Classifier on SpamBase.arff
Transform C4.5 RULES to SPARQL Queries
Execute SPARQL queries on BASE_SPONTO
User
User filters which C4.5 rules to apply
Identify instances which match C 4.5 Rules criteria and are misclassified
Output RDF Nodes which represent mis-classified SpamBase instances
Update (Delete / Fix / Modify Instance weight) Nodes
Ham
Spam
User
Protege

Figure 4-7: RDF assisted classification process.

4.6 Experiments and Results

As presented in the previous sections (*refer to Figure 3-6: Sequential SMO training time and Figure 3-7: Accuracy of the Sequential SMO*), the initial MRSMO prototype tries to keep classification accuracy close to global SVM optimization solvers by adopting effective strategies for the respective global weight vector and bias computation. The prototype processes and optimizes each data fragment in parallel using respective map operations. The output of each map operation reflects the partial weight vector for the localized data fragment. The single reducer sums up the respective partial weight vectors to compute the final global equivalent. It also works out the final bias threshold by averaging each respective partition's bias output.

The baseline SMO algorithm is an inherently sequential algorithm making use of single global data structures. The decomposed distributed version on the other hand employs a number of separate support vector machines based on the specific file splits, or rather training sets. This is what introduces the accuracy discrepancy between the sequential and distributed variants of the algorithms as implemented.

In this respect, the base training sets are supplemented with additional intelligence through end user contribution and feedback. The distributed SMO algorithm (*MRSMO*) is applied for classification of SpamBase test instances and outputs a set of pre-computed values which are the respective weight and bias vectors that reflect the SVM classifier. The pre-computed model is subsequently employed to perform further classification – the actual spam filtering - therefore bypassing the computationally taxing support vector computation required for training the model.

For the classification experiments carried out in this context, the SpamBase [191] dataset was employed once more. In every experiment carried out herewith, the entire feature set was employed during respective tests. Evaluating the performance of the classifier using pre-computed weights and bias constituting the classification model and selecting a set of 1267 random instances for testing from Weka's evaluation feature yields the figures presented in Table 4-5.

Table 4-5: Random instance classification

Outcome	Value
Correctly Classified Instances	1072 (84.6442 %)
Incorrectly Classified Instances	194 (15.3558 %)
Relative absolute error	50.708 %
Root relative squared error	62.6923 %
Total Number of Instances	1267

The RDF based feedback loop presented in the previous section is performed on this instance data. Figure 4-8 portrays the percentage accuracy improvement across the number of file splits. There is an average of ≈ 5 % accuracy improvement overall, ranging from 1.7 % when the file splits are at the minimum, namely 4 chunks to a maximum of 7.5 % when the number of input files is 48.

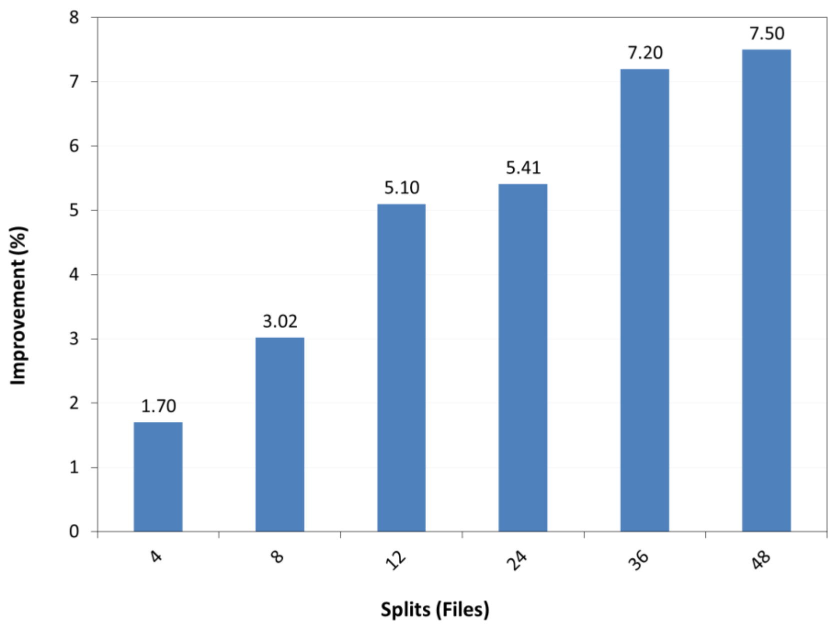


Figure 4-8: The impact of file splits on accuracy improvement

Figure 4-9 presents a comparative analysis of the rate of accuracy degradation between the RDF based intelligence supplemented approach (*designated X-Spambase*) and the original (*designated Spambase*) – the diagram shows that the former is significantly slower when increasing the number of file splits or rather the number of training input files – an aspect which influences the overall accuracy considerably in the original M/R SMO based model.

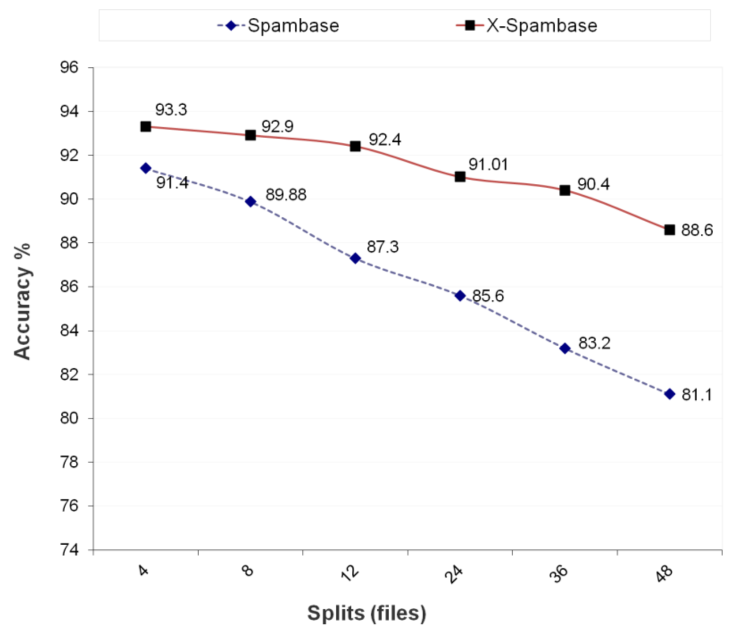


Figure 4-9: The impact of RDF augmentation on accuracy degradation

Based on an average accuracy improvement of 4.6% over the baseline distributed SMO, Figure 4-10 shows that using the RDF based feedback loop, the MapReduce based distributed SMO achieves a maximum accuracy of ≈ 99% and an average of 96%, which is better than the original sequential SMO [214].

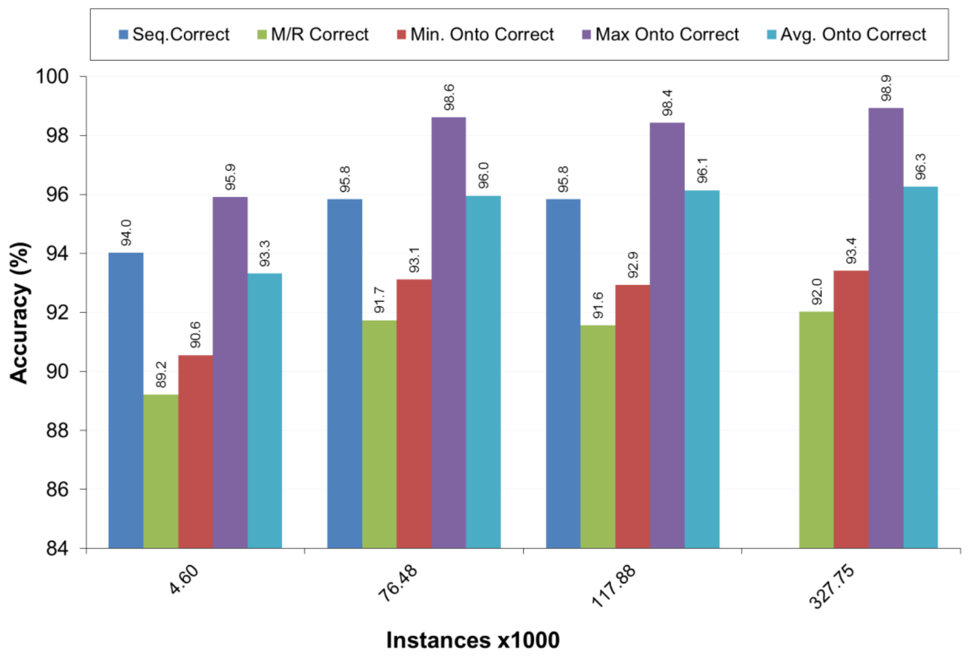


Figure 4-10: The accuracy of the RDF augmented MapReduce SMO

A large TREC data set [220] was transformed to a format similar to SpamBase for further evaluation of the RDF supplemented approach. The TREC corpus is constituted of about

75,000 messages out of which 50,000 are spam. An exploratory experimental assessment, based on approximately 1500 test instances, shows similar accuracy improvement over the original, distributed trained, MapReduce model as portrayed in Figure 4-11 and Figure 4-12 respectively, where 'Trec' represents the original accuracy and 'X-Trec' represents the corresponding RDF supplemented approach.

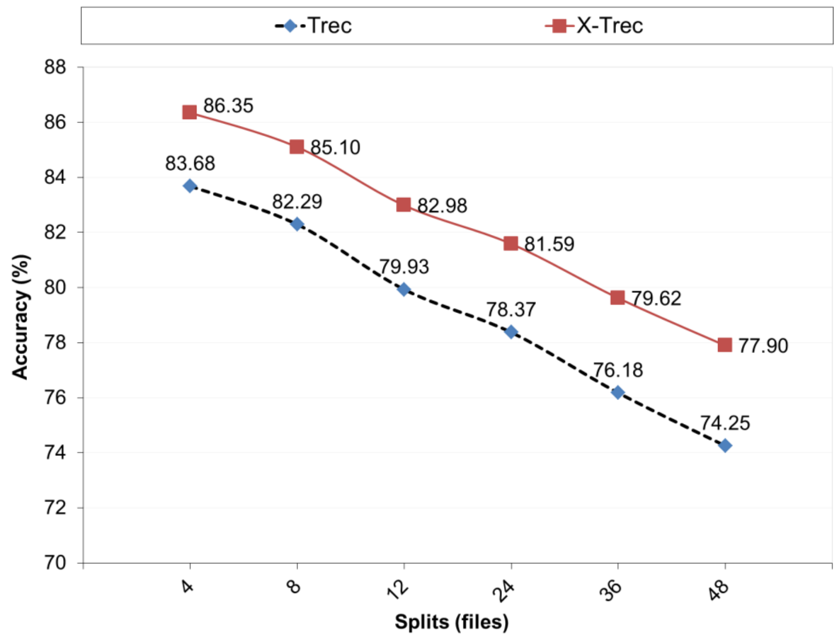


Figure 4-11: MapReduce SMO accuracy degradation with RDF augmentation-TREC

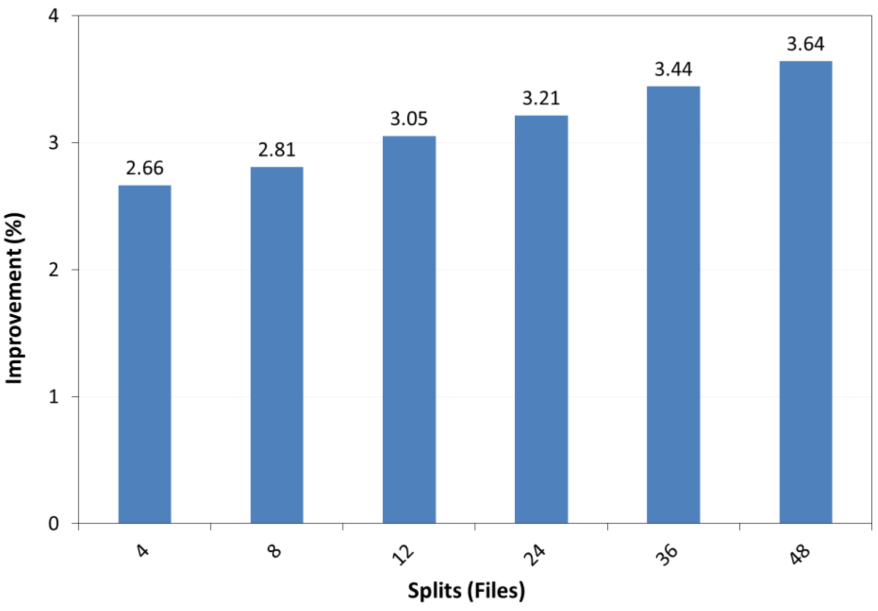


Figure 4-12: The impact of TREC file splits on accuracy improvement – TREC

4.7 Summary

To mitigate accuracy degradation in classification, in this chapter, an approach how the distributed SVM accuracy can be improved is presented. In contrast with the sequential SMO algorithm, the distributed version employs a number of separate SVMs using corresponding data file splits which degrades accuracy in classification. To mitigate this degradation, the base training data sets is supplemented with additional intelligence through an RDF based feedback loop. The SVM testing sets are transformed into a Resource Description Framework (*RDF*) graph representation. Misclassified instances are identified using automatically generated SPARQL [21]. Protégé [218], Jena [217] and Pellet [215] are employed for this, as well as the associated experiments. The supplemented intelligence is deployed to the original training data sets and the distributed SVM model re-computed. Experimental results indicate that the RDF based feedback loop improves the overall accuracy of the distributed SVM in classification by an average of 4.6 %. The effect of conventional bagging and boosting techniques on the performance of the distributed SVM is also evaluated earlier in this chapter. The chapter concludes with a discussion on related work in the context of the application of semantic and ontology based concepts for tackling spam filtering and classification.

CHAPTER 5 - Optimising Task Allocation in Heterogeneous Hadoop Cluster Environments

Beyond the actual classification schemes, the enabling infrastructure behind spam filtering also plays a critical role. The architecture(s), including hardware, software and networks, that are employed to perform typical anti-spam operations are commonly built using an ad-hoc approach. This regularly results in such architectures becoming quickly obsolete in terms of their effectiveness, or too costly to scale and keep up with the continuously changing spamming approaches. Such constraints, amongst others, lead to anti-spam setups that do not lend themselves very well to today's continuously evolving spam filtering capacity requirements and which mandate multi-platform support, scalability and extensibility out-of-the-box.

In this chapter, a heterogeneous aware M/R task to node matching and allocation scheme is explored and proposed. The original MRSMO work is extended with the intent to come up with a 'turnkey', high performance, scalable spam filter training approach able to capitalise on heterogeneous cloud based computing environments. A cloud computing centric virtual stack is configured with the proposed M/R task allocation/node matching scheme to complete the objective of providing a commodity based, high performance, scalable, spam filter training and classification stack.

The rest of the chapter is organized as follows. Section 5.1 provides a discussion on related work and Section 5.2 introduces briefly the challenges surrounding MapReduce in a cloud computing construct. Section 5.3 and 5.4 presents the design and implementation of gSched, the heterogeneous aware task to node matching and allocation scheme for Hadoop MapReduce proposed herewith. Section 5.5 presents a comparison and evaluation of gSched in a number of contexts, showing the improvements of the proposed approach 5.6 closes the chapter by summarising its more salient perspectives.

5.1 Related Work

The field of task allocation and scheduling in general is very well studied. Heuristics play a very important role in this context. Numerous studies evaluate the key characteristics of both static and dynamic heuristics [221][222][223]. Heuristics frequently underpin workflow scheduling research work, including that which considers different processing capabilities [224] [225].

In a typical Map/Reduce scenario, there are no strong task precedence constraints and consideration requirements beyond the order of Reducers needing to start after the associated Map phase. Basically, Map tasks do not have any order in isolation, and neither do Reduces. There is also no real intrinsic priority in terms of the individual Maps and Reduces within particular jobs. These considerations introduce the opportunity for relaxing typical directed acyclic graph heuristic scheduling dependencies and computational overhead associated with their generation and maintenance.

Purely from a Hadoop MapReduce performance improvement perspective, whilst different studies and standpoints have been considered [226][227][228][229], scheduling is still the more prevalent. Different scenarios obviously mandate different techniques. However, any scheduling scheme, irrespective of type, introduces specific challenges, complexities and thus processing requirements.

It is thus imperative to try to establish a good balance between accuracy and performance [230]. To try to achieve this, the work "*Dynamic proportional share scheduling in Hadoop*" [231] focuses on capacity distribution. Within specific processing windows, users are allocated slots for task processing on a proportional time share basis based on priorities, pre-emptively. This is somewhat similar to Hadoop's default scheduler which is also capable to 'time-box' tasks. However, this approach does not try to exploit underlying heterogeneous capabilities. The latter challenges can also be attributed to the work presented in [232]. The assumption of a homogenous environment here is also extended to the processing unit cost. In a cloud computing context, these assumptions break down - heterogeneous resources consideration is a fundamental perspective in this context. increasing evidence shows that heterogeneity problems must specifically be tackled in MapReduce frameworks [233].

Hadoop MapReduce 'per-se' is not specifically bound to any degree of homogeneity in terms of computing resources. There are numerous approaches of how heterogeneity can be tackled. The consideration for workload types and queues are popular research avenues

[234], as well as other techniques which focus on specific application areas [235][236]. In contrast with the view presented in [231], the opportunity to employ simplified task matching and allocation approaches is believed to be a good opportunity to consider in this respect, especially in modern, metamorphic, cost sensitive cloud computing based environments. Formulating a good approach which takes into consideration the capitalization of heterogeneous environments in the balance of cost is believed to still be an open research question. In this respect, in this work the application of a machine learning scheme for primary task characteristics classification and a distance vector node similarity approach for exploiting heterogeneity is explored.

In [237], the authors adopt a modified approach to the default Hadoop MapReduce speculative execution strategy. A number of resource allocation policies are described which are intended to “steal” unutilized slots and allocate them to specific running jobs. However, this can lead to contention in terms of subsequent tasks being starved of processing slots which are now occupied by tasks which are allocated such “stolen” slots. The process for terminating such tasks can actually lead to additional maintenance overhead and subsequently cost. Under specific circumstances (*heavy cluster load/usage for example*) this will also hamper and limit the overall effectiveness.

In [238], the author takes a different and innovative approach based on the baseline Hadoop fair scheduler. When jobs with higher computing rates become available, the scheduler automatically selects these to improve overall performance. This is done by exploiting the underlying heterogeneity better without starving other jobs from computing time. The scheduling scheme presented in this dissertation work tackles this notion of task throttling from a different perspective. Rather than using the computing rate as a baseline influencer for job/task scheduling selection, the degree of compatibility between the machines and tasks features is employed. This is achieved by recording key machine and already executed tasks characteristics. The degree of compatibility is computed via distance vector metrics – the closer the similarity the more comparable the capabilities for nodes.

In [239], LATE – a longest approximate time to end approach is employed to throttle tasks before execution. The approach proposed gives priority to tasks which will impact response time mostly. These are scheduled immediately on the fastest node of the processing cluster.

This is similar in part to this work. However, in this research work, a machine learning based approach to establish the task bias, based on its characteristics' is employed. This is done to with the intent to establish a good task to node processing matching and association baseline. Furthermore, in contrast with [239], for the proposed approach the basic original speculative execution strategy is kept. A priori however, the proposed scheme focuses on trying to limit the number of erroneously launched speculative tasks on potentially non-optimal nodes. In a cloud computing scenario this can reduce costs influenced by performance and resource capability inconsistencies in highly multiplexed, virtual resources based environments.

Unless adequately governed, the overhead of handling errors and stragglers can offset the potential performance gain in a cost sensitive cloud computing context. Again, in the proposed approach, a machine learning and distance vector based approach to objectively identify which nodes can reduce the occurrence of unnecessary speculative execution is used. gSched continuously monitors cluster capabilities by re-profiling nodes on a regular basis. This allows gSched to handle the challenge related to the highly multiplexed, virtual resources based environments runtime capabilities which change over time. Furthermore, as already indicated, the slight relaxation of typical Hadoop MapReduce data locality scheduling bias is considered in conjunction with heterogeneity. The decision to schedule local data fragments for processing is taken in conjunction with an a-priori establishment of whether the current node (*with local data*) is appropriate for the task type.

5.2 Heterogeneous MapReduce Environments

The ability to maximize and exploit heterogeneous resources for MapReduce processing in cloud computing environments has become an increasingly compelling scenario for processing large scale data sets and / or compute intensive workloads. MapReduce scheduling can be considered as a somewhat different, perhaps simplified, flavour of workflow scheduling - the primary order constraints related to the execution of the Reduce phase after the respective/associated Mapper rather than specific atomic task operation order. Hadoop [30] is considered one of the more popular MapReduce implementations even in a Cloud computing context. The default Hadoop FIFO scheduler employs a straightforward yet effective strategy to allocate tasks. Hadoop also supports an alternative fair (*originally developed by Facebook*) and capacity (*originally developed by Yahoo*) scheduling schemes. However, neither considers the heterogeneous perspective explicitly. By default, when a node has an empty task slot, Hadoop selects a task for it from one of three categories. First, any failed tasks are given highest priority. This is done to detect when a task fails repeatedly (*for example due to a bug*) and stops the job. Secondly, non-running tasks are considered. For maps, tasks with data local to the node are chosen first. Finally, Hadoop looks for a task to execute speculatively. Speculative execution is primarily intended to execute long running tasks on more than one node. Purely from a performance perspective, this provides a degree of control on stragglers - nodes which are relatively slow performing.

On this premise, Hadoop's default MapReduce scheduling approach is still regarded as somewhat inefficient for heterogeneous environments. It assumes a degree of homogeneity which, in modern computing scenarios, especially in cloud computing environments, is not common. Furthermore, for the latter, the underlying resource capabilities can actually vary during job execution. The differences and variation of these capabilities must be taken in consideration in order to optimize any task allocation strategy from a cost and performance perspective.

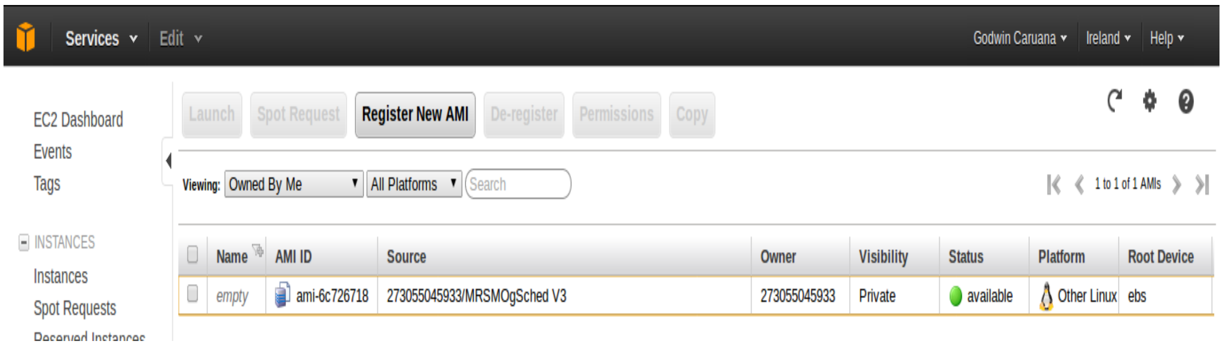
5.3 The Design of gSched

In a cloud computing environment, numerous 'infrastructure' scheduling schemes for the actual computing resource provisioning are in place. Any higher-level task matching and allocation scheme thus need to take into consideration these elements from ground up. Furthermore, they also need to ensure that there is the required degree of fairness, mitigate starvation scenarios as well ensure efficiency. These constraints make scheduling in a heterogeneous environment challenging - the overall scheduling strategy needs to be adaptive and dynamic.

For this work, a specifically configured Amazon Machine Image – AMI was configured and packaged. The AMI represents a virtual operating environment, complete with operating system and specific software packages and configuration. This was done to be able to have the capability to launch multiple instances of the image to shape the Hadoop MapReduce cluster, fully configured with the basic as well as the proposed (*gSched*) task to node matching and allocation scheme. This provides the opportunity to deploy a virtual, MapReduce based, spam filter training architecture in cloud computing construct easily. The baseline configuration of the virtual 'appliance' is described in Table 5-1. Figure 5-1 shows the specifically configured AMI instance that thus encompasses this baseline configuration. The AMI package is exportable to mainstream virtualization formats via the Amazon EC2 API tools - thus can be used in most virtualization contexts.

Table 5-1: AMI Virtual Appliance software baseline

Software Environment	
SVM	Weka 3.6.0 (SMO)
O/S	Ubuntu 12.10
Hadoop	Hadoop 0.20.205
Java	JDK 1.7



	Name	AMI ID	Source	Owner	Visibility	Status	Platform	Root Device
	empty	ami-6c726718	273055045933/MRSMOgSched V3	273055045933	Private	available	Other Linux	ebs

Figure 5-1: Specifically created & configured Hadoop 0.20.205 Amazon EC2 AMI Stack

5.3.1 Design

Implemented within a Hadoop context, gSched is intended to try to exploit heterogeneous capabilities in a cost effectiveness construct. Various elements influence MapReduce 'processing' costs in a typical cloud context. For this work it is assumed that capital expenditure (*capex*) can be "ignored" (*hardware, software etc.*). Thus capex $\rightarrow$ 0. The opex perspective, operational expenditure, includes the costs for the resources acquired - R - (*which includes storage, memory and processor etc.*) can be typically represented as follows:

$$c = \sum_{t=1}^{T}\sum_{n=1}^{N} R(n) * u \qquad \textbf{[5-1]}$$

Where:

- c is the operational expenditure (opex)
- T is the time span for resource use
- N is the total number of nodes
- R is the resources leveraged/acquired
- n is a specific node
- u is the unit cost

gSched tries to establish a balance between minimizing the time span for resource use whilst maximizing the heterogeneous resources acquired (*i.e. with the constraints*), namely:

$$f\,(\mathrm{c}^*) = [\mathit{minimize}\ \mathrm{T},\ \mathit{maximize}\ \mathrm{R}] \qquad \textbf{[5-2]}$$

The proposed approach also takes into consideration that in real-world, large scale cloud based processing exercises, external storage, rather than locally attached, may be preferential for persisting data. Thus, the rank of Hadoop's MapReduce data locality scheduling bias is slightly relaxed. This also based on the rationale that in cloud based environments, replication is a costly measure [240]. Similarly, considerations for traditional inter-cluster network performance have also been relaxed [30]. Network performance in Amazon EC2 and S3 constructs for example, have been proven adequate for this type of task [241]. Obviously, approaches such as HDFS staging or using the actual Amazon EC2 storage can obviously be employed. However, these approaches are expensive. Data locality [240] is still capitalized upon, but this is considered in the context of heterogeneity for increased effectiveness.

An a-periodic, arbitrarily divisible task set scenario is assumed. The gSched task matching and allocation scheme is not concerned with task order or precedence in general. Map jobs are independent of each other. The sole precedence consideration is that Reducers start after Mappers are finished. The actual matching and allocation approach is based on a number of processes. An Estimated Completion Time (*ECT*) based technique for establishing a priori an indicative processing time for executing a task *T* on a node *N* with processing characteristics *C* is employed. This is done by ‘profiling’ tasks and establishing the respective processing bias (cpu, disk I/O). The same approach is adopted to establish the degree of ‘similarity’, and thus capabilities, of participating nodes.

A machine learning scheme is employed to establish whether the task type or rather its bias, is CPU or I/O bound. Whilst data skew can introduce task processing characteristic differences [242], the proposed approach is based on the assumption that under most circumstances, the individual tasks of a specific MapReduce Job are reasonably the same in terms of their CPU, Memory and I/O requirements. More specifically, in a typical Hadoop MapReduce scenario, job composition is constituted of a number of logically 'equivalent', arbitrarily divisible tasks that require scheduling. Thus, if a job J is constituted of tasks, {t1….t4}, t1≈ t2≈ t3≈ t4.

ECT is commonly used to baseline and subsequently project or infer important task information [221][243]. Various generation methods are used to generate the ECT matrix values [244]. In this work, an inconsistent ECT technique is employed, benchmarking as well as recording the characteristics of a node and inferring, based on machine heterogeneity considerations, a plausible task processing time for other participating nodes. This allows gSched to deduce an estimated, relative time to process a task on nodes beyond the one actually employed for processing.

Through the machine learning scheme, the task bias is identified and subsequently allocated (*the task*) to a specific machine (*node*). When the task is processor bound, the machine which has the ‘shortest’ relative ECT time is selected (*offset from fastest node is actually configurable*). If the task is not CPU bound, the algorithm allocates the task to a machine selected by minimizing the ECT mean for the task ready for scheduling. The focus and basic distinction being CPU and IO bias is based on the premise that it is in general, the former is the most ‘expensive’ resource in an on-demand provisioning computing scenario. It is also based on the simulation results outcome studied from the HSIM simulations presented in section 3.4.6 in terms of MapReduce behaviour and performance impacting characteristics.

In this proposition, a basic set of characteristics (*signature*) which reflect the participating nodes capabilities are recorded. These include the number of processors, the available physical memory, available disk space, the CPU performance and the total physical memory.

For each job, a configurable minimum number of tasks are first executed. These initial set of tasks are scheduled using the default FIFO approach adopted by Hadoop MapReduce. The 'task shape'- a set of key characteristics which describe each task is also recorded. These are the time the task the time takes to execute, the physical and virtual amount of memory employed by the task and whether the tasks executed was primarily CPU or I/O bound. The rest of the tasks from the same Job are then scheduled as described in the following paragraphs based on the ECT information available.

5.3.2 ECT Estimation

The objective of the ECT estimation model is to infer an estimated compute time of a task on participating nodes. This is based on node similarity, performance and actual processing times (*known a priori, of a typical task from the same Job*) of specific, logically 'equivalent' tasks on specific nodes.

The general performance characterises of a node is a function of the capabilities of a number of core components including processor speed, disk I/O speed and memory size (see Equation 5-3). For the actual execution of a process (*task in this particular context*), a number of additional perspectives also influence general performance, including virtual memory for example.

During the Hadoop MapReduce cluster start-up, as well as periodically (*configurable parameter*), gSched profiles the participating machines capabilities and characteristics. This is performed to be able to mitigate to an extent the performance changes the underlying, highly multiplexed, environment performance characteristics. In order to establish the estimated compute time of a node, for this proposition these key characteristics are represented as a vector. The general vector space model is a simple yet effective technique to store representing information. Each node has an associated performance characteristics vector associated with it, created during the respective profiling process.

Cosine similarity is an effective scheme for establishing how similar two vector are – perhaps more importantly is that it is also computationally simple to perform such an operation. Whilst the vectors are non-sparse, their dimensionality is small and the overall complexity of the similarity computation is *O(n)*. This is an important facet to consider and one which influenced the decision to undertake such an approach. The overall computation time taken to actually establish the "best" node to task match (*gSched overhead*) has to be kept small. In this approach, the node similarity, $n_{Similarity}$ is based on the core characteristics ($mSig_n$) of the underlying node(s) participating in the MapReduce cluster:

$$mSig_n = [\text{mips, number of cores, cpu frequency, io speed, disk space, physical memory, virtual memory...}] \quad \textbf{[5-3]}$$

$$n_{Similarity} = \sum_{i=1}^{n}(nSig_i \text{ x } nSig_\text{n}) / (\sqrt{\sum_{\text{i}=1}^{\text{n}}(nSig_\text{i})(nSig_\text{i})} \text{ x } \sqrt{\sum_{\text{i}=1}^{\text{n}}(nSig_\text{n})(nSig_\text{n})}) \quad \textbf{[5-4]}$$

Where:

$mSig_n$ is a node characteristics profile vector of each node

$n_{Similarity}$ is the cosine similarity value between the two nodes being compared

A number of additional components are employed for the actual execution time (ECT) estimation. These are:

1. The node performance/benchmarks ($node^A_{Bench}$, $node^B_{Bench}$) – which is part of the node signature mSign, the node characteristics (*nSig [nodeA] and nSig[nodeB]* and
2. The original task time – oTT (*the time for one task of a Job from which a number of tasks were actually executed and thus actually timed*), and the node similarity nSimilarity (*nodeA, nodeB*).

There are 3 main scenarios to consider for this ECT estimation approach, namely:

If $node^A_{Bench}$ is the 'same' as $node^B_{Bench}$

$$\text{ECT} = \bar{\mathcal{X}}\,(\,(oTT * 1/n_{Similarity}\,(nSig\,[node^A],\ nSig\,[node^B]))\,,(oTT * n_{Similarity}\,(nSig[node^A],\ nSig\,[node^B]))\,) \quad \textbf{[5-5]}$$

If $node^A_{Bench}$ is *'better'* than $node^B_{Bench}$

$$ECT = oTT - ((node^A_{Bench} - node^B_{Bench}) * oTT / n_{Similarity}) \quad \textbf{[5-6]}$$

If $node^B_{Bench}$ is *'better'* than $node^A_{Bench}$

$$ECT = oTT + ((node^A_{Bench} - node^B_{Bench}) -1) * oTT / n_{Similarity})) \quad \textbf{[5-7]}$$

The CPU performance benchmark is base-lined using a timed execution of a Fibonacci sequence computation. Correspondingly, the I/O benchmark is established via the timed execution of a set of write and read operations on a pre-specified random data content file.

5.3.3 Task to Node Matching and Allocation

For this work, it is assumed that the underlying network and its influence would be relatively stable. For future work it is intended that this consideration is explicitly included as part of the matching and allocation scheme. The estimated completion time matrix is updated with the actual processing times and task characteristics on the nodes the tasks were executed. The estimated time that the same task (*with its associated*) shape (*or characteristics*) will take on the nodes which the task was not actually executed on is then inferred. This is based on the variation of node characteristics difference using a distance vector scheme.

For the remaining tasks, rather than allocating the tasks to nodes with empty slots immediately, gSched initially throttles (*hold*) back these tasks, in part similar to [239] for a maximum number of times (*configurable*). gSched establishes whether the task types pertaining to the specific job exhibit CPU bias or otherwise using the naive Bayesian classification scheme. Machine learning has been applied in various contexts including in coming up with scheduling strategies which evolve and self-tune to the scenario at hand [224][245]. Bayesian classifiers are considered very effective where inference is required from data which is not necessarily of the highest level of accuracy representation.

Based on whether the tasks exhibits CPU or IO bias, gSched selects an available node which is more congruent with the characteristics of the job/tasks at hand. If any node is not allocated a task for a number of ‘attempts’, the task is subsequently scheduled ‘forcefully’. This is applied in order to ensure that there are no idle resources for uncontrolled periods simply because the matching scheme did not identify a good node / task combination candidate.

5.4 gSched Implementation

The overall process involved in the task selection and scheduling is represented graphically in Figure 5-2.

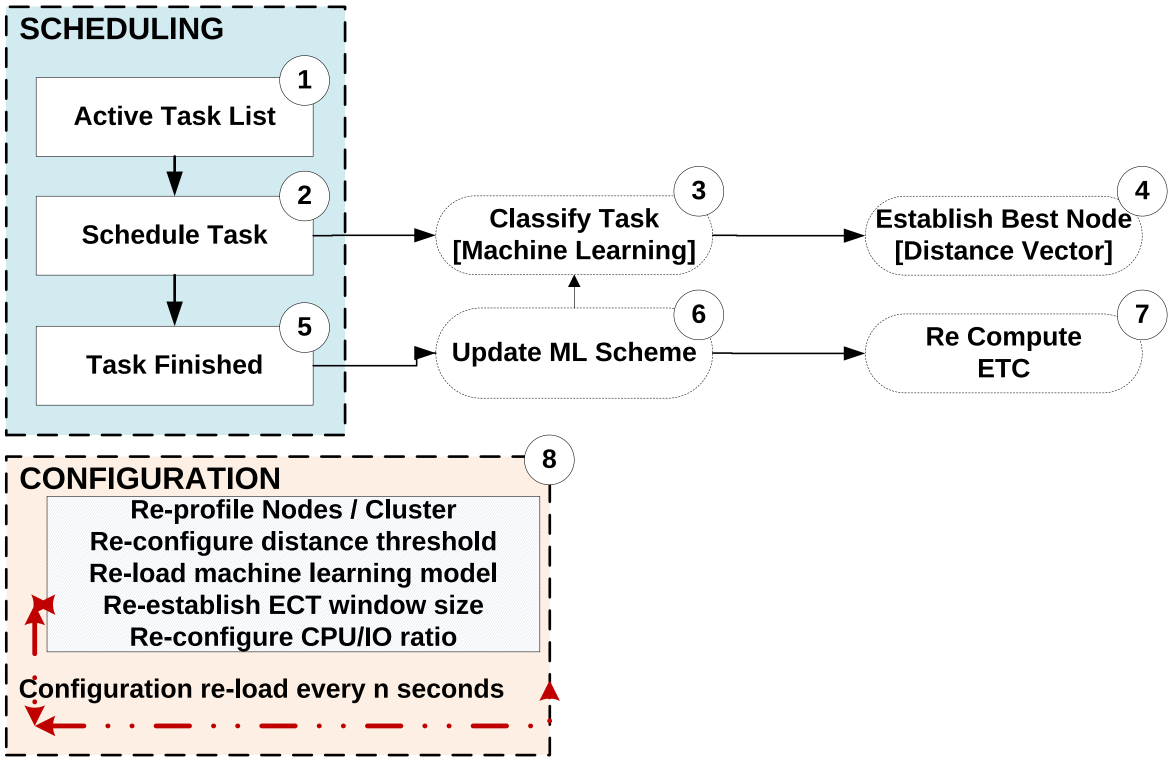

SCHEDULING
1
Active Task List
2
Schedule Task
5
Task Finished
3
Classify Task [Machine Learning]
4
Establish Best Node [Distance Vector]
6
Update ML Scheme
7
Re Compute ETC
8
CONFIGURATION
Re-profile Nodes / Cluster
Re-configure distance threshold
Re-load machine learning model
Re-establish ECT window size
Re-configure CPU/IO ratio
Configuration re-load every n seconds

Figure 5-2: The design of gSched

This next sections provides an overview of the core algorithms constituting gSched, represented via the following pseudo code.

- $C_{nodeCount}$ is the number of nodes within the cluster
- C_{sig} represents the cluster signature
- N represents a node
- E represent the node (n) capability
- f_{slots} represents the number of node free slots
- $f_{maxslots}$ is the nodes maximum free slots
- NodeWithFreeSlots[] is list of n(odes) with fslots > 0
- J represents the M/R Job
- J_{tot} refers to the total M/R Jobs
- J_t is the M/R Task within the Job (J), such that t > 0
- J_{ttime} refers to the time to execute J_t
- T_{time} equals the time to execute a task within a job (J)
- $J_{numtasks}$ refers to the number of tasks in Job (J)
- $J_{tottasks}$ is the total number of tasks in Jtot

- MLScheme represents the Machine Learning Scheme
- $n_{sig(n)}$ is the Signature of Node (n) with E capability
- J_{tshape} refers to the Task Node Utilization Vector
- $ECT_{Jtshape}$ is ECT signature for Task Node Utilization
- ECT refers to the Estimate Time to Compute Matrix
- T_{Class} represents the Task Class based on ML_{Scheme}

Algorithm 5-1: Cluster Start-up (& Re-profiling)

```
(Step 8, Figure 5-2)
1   ∀ i = 0  $C_{nodeCount}$
2           profile machine E(n[i])
3                   generate node(i) signature $n_{sig}${
4                           diskSpace, MIPS, numProc, MEM }
5   store profile configuration $C_{sig}$
```

Algorithm 5-1 is triggered at cluster start-up (*and during any re-profiling iterations – Step 9, Figure 5-2*). It identifies and records the key characteristics of participating nodes, including the disk space, CPU processing capabilities (*MIPS*), number of processors and memory etc. The associated cluster profile functionality (*Step 9, Figure 5-2*) allows gSched to gain insight and track changing runtime capabilities which occur in highly multiplexed, virtual resources based environments.

The ability to re-profile participating nodes, as well as changing its configuration at runtime allows gSched to dynamically adapt to varying underlying cluster capabilities over time.

Algorithm 5-2: Task Node/Matching and allocation

```
(Step 2, Figure 5-2)
1:    assuming $J_{t1} \approx J_{t2} \approx J_{t3} \approx J_{tn}$
      ∀$J_t$ = 1 to $J_t = f_{maxslots}$
2:    select n ← AVG relative time from ECT.
      if no ECT info available, select randomly
      allocate 1st set of ($J_t$ = 1 to $J_t = f_{maxslots}$depending on
      NodeWithFreeSlots $f_{slots}$ and $f_{maxlots}$ on selected n
      where $f_{slots} > 0$
3:    generate/Update ECT Matrix
4:    ∀ $J_t > f_{maxslots}$
5:    Task Vector = $J_{tshape}$
6:            classify Task TClass using MLScheme
7:            on selected Jtshape
```

```
8:    **schedule** remaining tasks, via
9:    **assign** task $J_t$ to $n_nM_{sig}$ (n)
10:   **where** $n_nM_{sig}$ (n) $f_{slots} > 0$
11:   **and**
      **if** TClass = {0} (cpu bound)
              select n =  Min+1 (m1Msig (1) - $m_nM_{sig}$ (m)) for
              ECT_Jtshape
      **else** select n = Minimize ($\bar{\mathfrak{T}}$(m1Msig (1) - $m_nM_{sig}$ (m)) -
              ECTtime) for ECT_Jtshape
              mark $n_nM_{sig}$ (n) $n_{slot}$ in use
```

Algorithm 5-3 represents the key elements of the allocation scheme. There are two scenarios. Where the number of tasks processed from a job *J* is less than an arbitrary number, i.e. there is no visibility of what type of task, in terms of characteristics, is intended for scheduling, gSched selects the node with an average relative time from the ECT. If there is no information in the ECT, the node is selected randomly. On task finish, gSched updates the ECT table accordingly (*Steps 1, 3 and 6, Figure 5-2*)

On the other hand, where visibility of tasks type (*shape*) is already available for a particular job, the respective task is classified, via the machine learning scheme, as CPU or I/O bound (*biased*) (*Step 2 and 4, Figure 5-2*). The relevant, and inferred, participating node processing times for the tasks already processed within the same job are then acquired.

If the task is CPU bound, the fitness function or node selection scheme allocates the task to the available node with the potential quickest processing time for the task at hand - also taking into consideration whether the node has free task slots available. Whether it is actually the best performing or a top ranked one within a range is configurable - gSched can also modify it according to the runtime behaviour of the underlying cluster based on node allocation contention.

The same general approach is applied to the I/O bound tasks. This time however, the scheme establishes which available node has the better I/O performance characteristics for task assignment.

This approach allows task assignment throttling to establish a good candidate combination of task and node, based on the current cluster profile (*Step 6, Figure 5-2*).

Algorithm 5-3: Task Finished

```
(Steps 6 and 7, Figure 5-2)
1:  re-compute ECT using end of task information
2:      infer node times from current using Algorithm 5-4
3:      update MLScheme with:
4:          Machine Signature $M_{sig}(n)$
5:          Task Signature (Mean Task Info) $Jt_{shape}$
6:          $Jt_{time}$ Task Time
7:          Configuration Signature $C_{sig}$
8:          CLASS { CPU Bound {0}, IO Bound {1}
```

Algorithm 5-3 performs the necessary housekeeping to update both the ECT table (*calling Algorithm 5-4*) and updating the machine learning scheme with the newly learned information of the finished task (*Step 6 and 7, Figure 5-2*). This will be used as a new training instance – Algorithm 5-5.

Algorithm 5-4: ECT Table Generation

```
(Steps 7, Figure 5-2)
1   ∀ i from 0 to $Cn_{odeCount}$
2:   if n[i] != n[Jt]
3:      $D_{min}$ = n[Jt].nshape *
        (||$M_{sig}$(n)[Jt.$n_{shape}$] ||$M_{sig(i)}$[n(i).$n_{shape}$] || cos Θ)
4:      $D_{max}$ =  n[$J_t$].$n_{shape}$* 1 / (||$M_{sig}$(n)[Jt.$n_{shape}$] ||
        $M_{sig(i)}$[n(i).$n_{shape}$] || cos Θ)
5:      n[i].$n_{shape}$= ($\bar{\mathcal{T}}$ ($D_{min}$, $D_{max}$))
6:   Else
7:      n[i] =  n[Jt].$n_{shape}$
```

The ECT table generation uses a straightforward approach – Algorithm 5-4.

Based on the actual time of a finished task, task characteristics and node characteristics, the potential time the task will take on the other nodes where it has not been scheduled (*Step 8, Figure 5-2*) is inferred using a distance vector scheme.

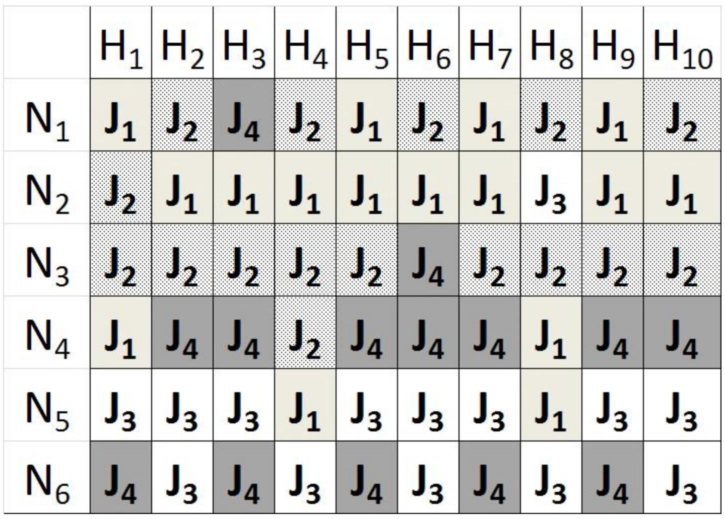
Algorithm 5-5: Machine Learning Scheme

```
(Steps 6, Figure 5-2)
input: Finished Task Information (Algorithm
output: Classifier
```

```
1:    Training Scheme
          if $Jt_{time}$ (CPU)>CPU_RATIO
          then CLASS {0}
          else CLASS {1}
2:    Classifier
      $C = P(C|F1, \ldots\ldots Fn)$
      $= \frac{P(C)P(F1, \ldots . Fn|C)}{P(F1 \ \ldots, Fn)}$
      C= $\frac{\operatorname{argmax} P\ (CLASS)}{CLASS} * \prod_{i=1}^{n} P\ (Jt_i\,|\,CLASS)$
      where CLASS = CPU or IO bound
```

The machine learning scheme adopted (*which can be changed - this is intended to be converted to a plug-in based architecture for future work*) is a Naïve Bayes. Algorithm 5-5 takes the training instances added by Algorithm 5-4 and re-trains the model. The model is then subsequently used for further classification. The model is only re-trained when the Hadoop M/R job queue processor is idle and there are no Jobs in the queue - that is there is no runtime performance impact when jobs are in the queue and scheduled.

Figure 5-3 shows an example allocation schedule based on gSched, using the Hadoop heartbeat (*H_1 - H_{10} in Figure 5-3*) as a baseline. Job J_1 tasks are initially allocated to Node N_1. gSched establishes that the characteristics of N_2 are more suited to the tasks requirements of the job. J_1 tasks are however also allocated to N_1. This for a number of reasons - gSched will allocate underutilized Nodes with tasks. Each Node is also not allowed to 'refuse' the allocation of tasks more than a configurable number of times.



	H_1	H_2	H_3	H_4	H_5	H_6	H_7	H_8	H_9	H_{10}
N_1	J_1	J_2	J_4	J_2	J_1	J_2	J_1	J_2	J_1	J_2
N_2	J_2	J_1	J_1	J_1	J_1	J_1	J_1	J_3	J_1	J_1
N_3	J_2	J_2	J_2	J_2	J_2	J_4	J_2	J_2	J_2	J_2
N_4	J_1	J_4	J_4	J_2	J_4	J_4	J_4	J_1	J_4	J_4
N_5	J_3	J_3	J_3	J_1	J_3	J_3	J_3	J_1	J_3	J_3
N_6	J_4	J_3	J_4	J_3	J_4	J_3	J_4	J_3	J_4	J_3

Figure 5-3: Example gSched allocation

5.5 Experiments

gSched and the standard Hadoop FIFO scheduler are initially base lined locally using the TestDFSIO and MRBench benchmarks [30] on an experimental cluster.

This comprised of the nodes as described in Table 5-2.

Table 5-2: Initial experiment – Test cluster

Type	Number/Specifications	Role
Physical	1 – 1024 MB, 2 core, 500GB HDD	*Master*
Virtual	1 – 768 MB Ram, 2 Core, 256 GB HDD	*Slave*
Virtual	1 – 512 MB Ram, 1 Core, 256 GB HDD	*Slave*
Virtual	1 – 384 MB Ram, 1 Core, 120 GB HDD	*Slave*

5.4.1 Base-lining gSched

Figure 5-4 and Figure 5-5 portray the performance difference in this respect between the standard scheduler and gSched on the experimental cluster.

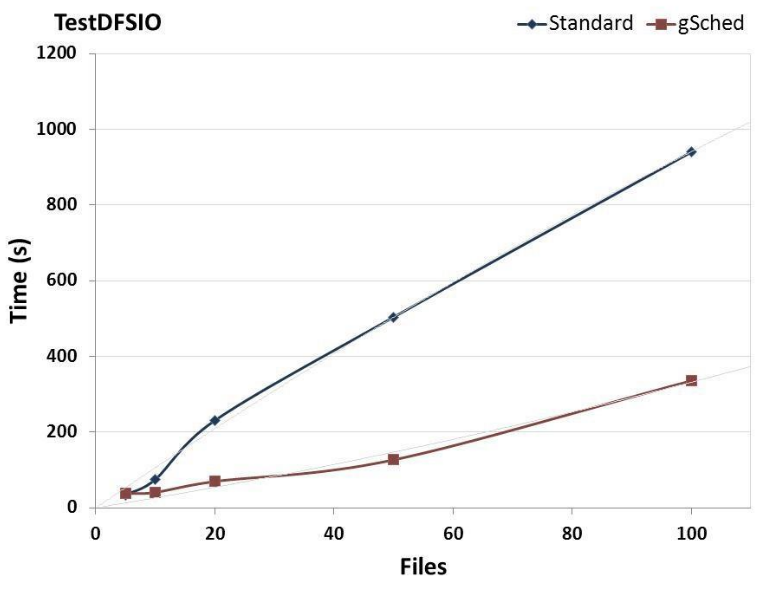


Figure 5-4: TestDFSIO - Standard vs. gSched

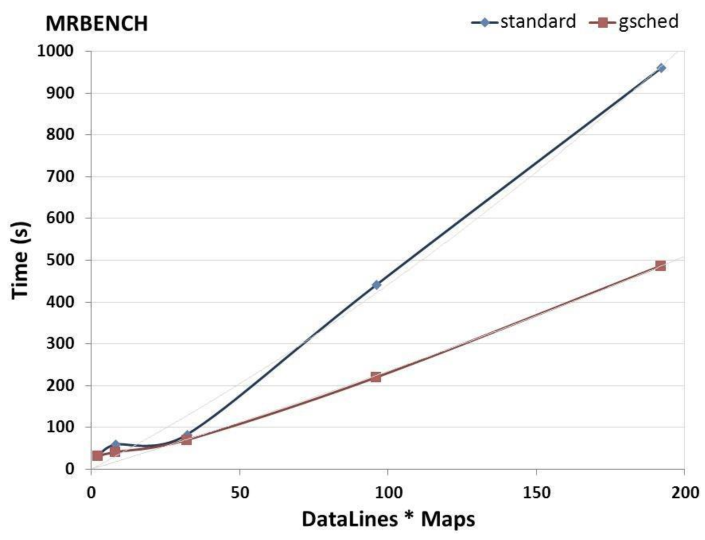


Figure 5-5: MRBench - Standard vs. gSched

For the TestDFSIO test, ≈ 10 Mb files were employed. For the MRBench test, the number of 'DataLines' was varied from 1 to 12 and the respective associated 'Maps' from 2 to 16. There is a marked performance improvement in favour of gSched in both tests.

In order to evaluate the performance of gSched in a typical cloud based environment, a number of additional experiments were carried out using Amazon's EC2 service [36]. At this stage, it is pertinent to note that it is difficult to clearly compare scheduling performance in a virtual resource provisioning platform context [246]. Several factors differentiate experimental setups and outcomes. These include the actual MapReduce operating environment (*OS*) scheduling strategy as well as the underlying HyperVisor multiplexing scheduling schemes.

Amazon provides a number of instance types for the provisioning of computing resources with different stated performance, specification attributes and characteristics [247]. First generation (*M1 series*) Amazon EC2 compute and S3 storage services were initially employed to carry out these evaluations. An initial base-lining experiment with 4 't1.micro' EC2 nodes using the TestDFSIO and MRBench tests was performed. This time, for the TestDFSIO test ≈ 50 Mb files were employed. The number of 'DataLines' for the MRBench was varied from 1 to 12 and the respective associated 'Maps' from 2 to 16. The expected outcome for this experiment was, given similar node characteristics (*4 't1.micro' EC2 nodes*), the performance of gSched and the standard scheduler should not vary considerably. Figure 5-6 and Figure 5-7 present the result of the experiment.

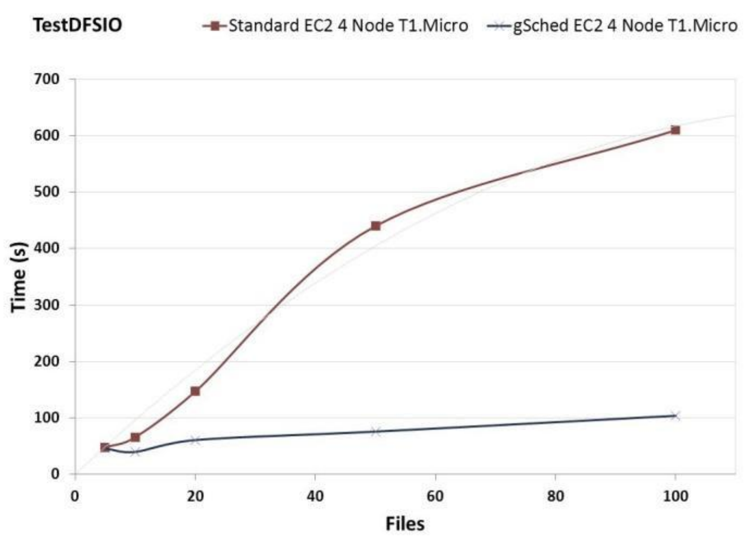


Figure 5-6: TestDFSIO - Standard vs. gSched – EC2 4 Node

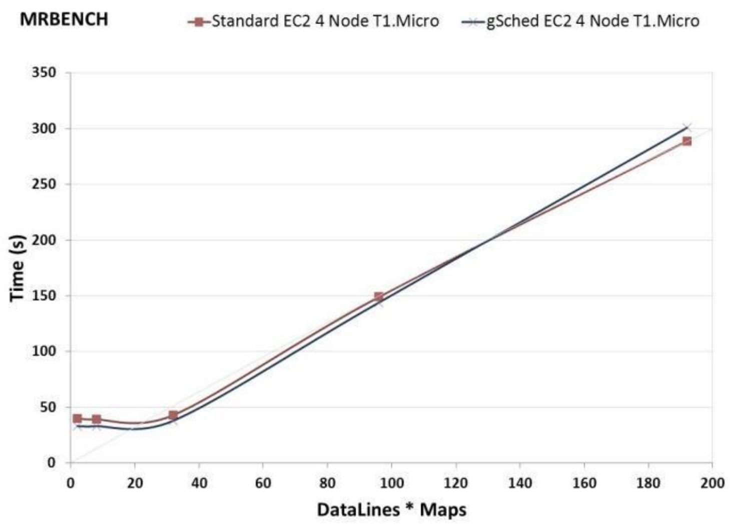


Figure 5-7: MRBench - Standard vs. gSched – EC2 4 Node

Figure 5-6 shows a significant difference between the standard and gSched performance for the TestDFSIO experiment. This was not expected – however, the discrepancy was traced to a large number of 'TaskRunner Child error's' for the standard scheduler execution test on the tested configuration. Consequently, the respective TaskTracker (*on node*) becomes 'blacklisted', in-turn effectively downscaling the cluster processing capabilities. On the other hand, the MRBench performance (*Figure 5-7)* is visibly very similar – this is by virtue of the absence of any opportunity for gSched to exploit heterogeneous performance differences for improved performance. In fact, the 4 nodes in this experiment are the same – EC2 t1.micro instances. Figure 5-7 also indicates that the performance overhead of gSched is not in any way significant.

5.5.2 Distributed SMO

In Chapter 2, various approaches for spam filtering were discussed. The use of the specific methods employed for spam classification, including SMTP and machine learning based approaches were explored. In the context of the latter, SVM based techniques have been proven effective for spam filter training and classification. However, SVM training is a computationally intensive process.

In Chapter 3, a MapReduce based distributed SVM algorithm, designated MRSMO, for scalable spam filter training is proposed. By distributing, processing and optimizing the sub-sets of the training data across multiple participating computer nodes, the distributed SVM reduces the training time significantly. An RDF semantics based feedback loop is subsequently employed to minimize the impact of accuracy degradation [214]. However, the overall training process is still considered computationally demanding. The possibility to further improve performance, in a cost effective fashion, make this specific scenario a compelling one to evaluate gSched's performance in this context. Using the Adult data set [195], in conjunction with the distributed SVM, MRSMO, an experiment to establish gSched's performance in comparison with the standard Hadoop scheduler in this scenario was performed. The simple Hadoop cluster setup described in Table 5-3 was employed. Figure 5-8, Figure 5-9, Figure 5-10 and Figure 5-11 portray gSched’s speedup and efficiency improvements over the standard scheduler for the respective SVM Polynomial and Gaussian SVM Kernel training tests as originally performed in [184].

Table 5-3: MRSMO with gSched – Test cluster

Type	Number	Hadoop Role
m1.medium	2	1 Master / 1 Slave
m1.small	2	Slave
t1.micro	2	Slave

For this experiment, the number of nodes as well as the number of MapReduce tasks were varied from 1 to 6 and 4 to 16 respectively. The Sequential SMO time for the Polynomial and Gaussian SVM training times are used as a reference baseline compared with the standard scheduler. Figure 5-8, Figure 5-9, Figure 5-10, Figure 5-11 show the speedup and efficiency of both kernels respectively in this experimental construct. The use of the Adult dataset, rather than SpamBase, was for the same rationale – namely to be able to compare the improvement in Speedup (*in seconds*) and Efficiency (*percentage improvement*) using the same baseline dataset employed for the Gaussian and Polynomial MPI comparison tests.

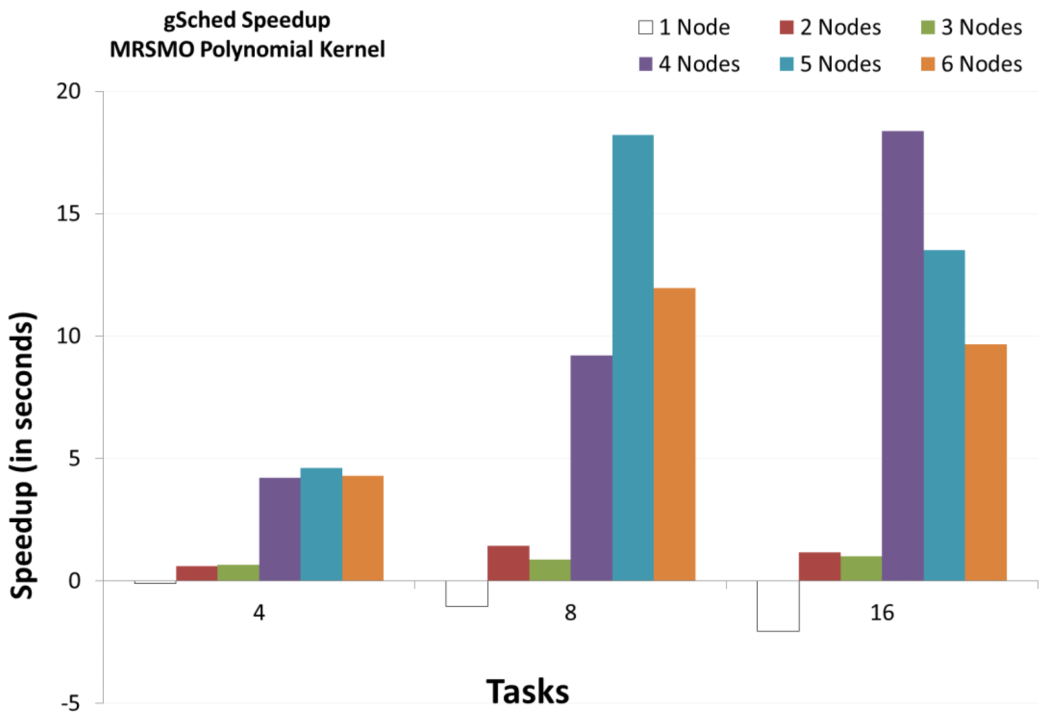


Figure 5-8: MRSMO - speedup (seconds) - Polynomial Kernel

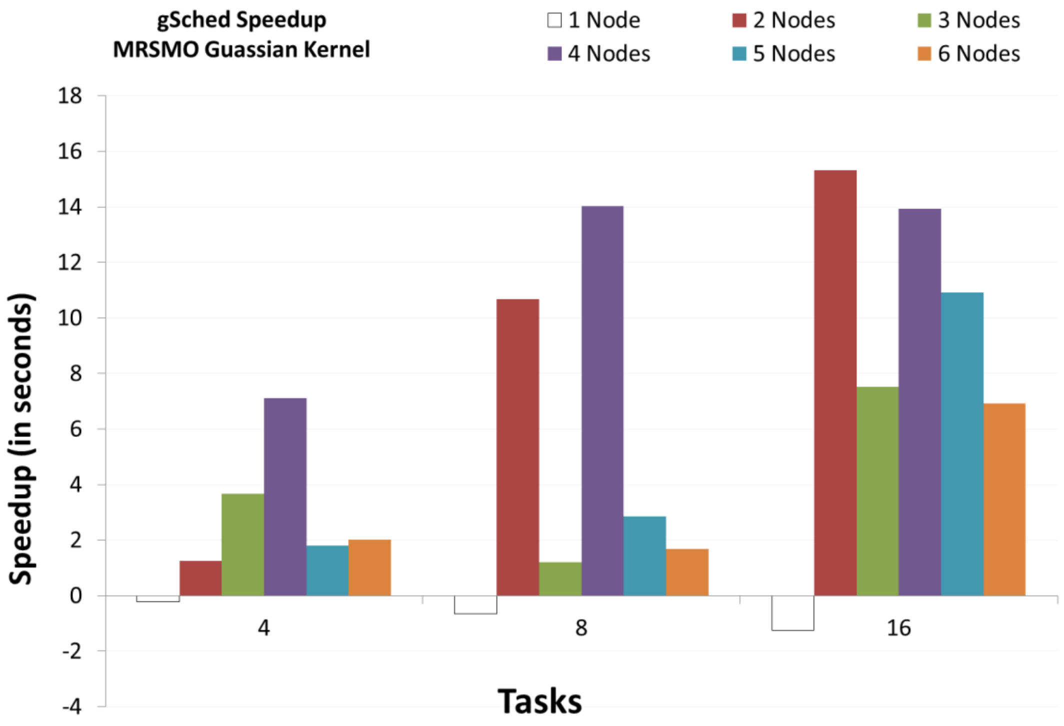


Figure 5-9: MRSMO – speedup (seconds) - Gaussian Kernel

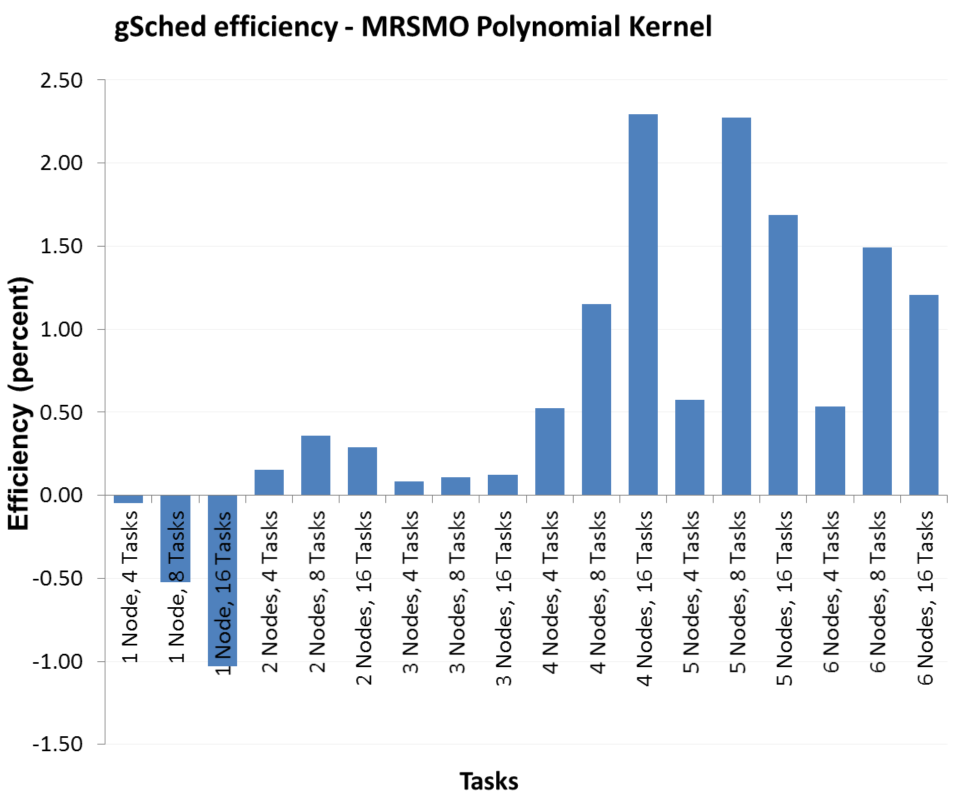


Figure 5-10: MRSMO - efficiency - Polynomial Kernel

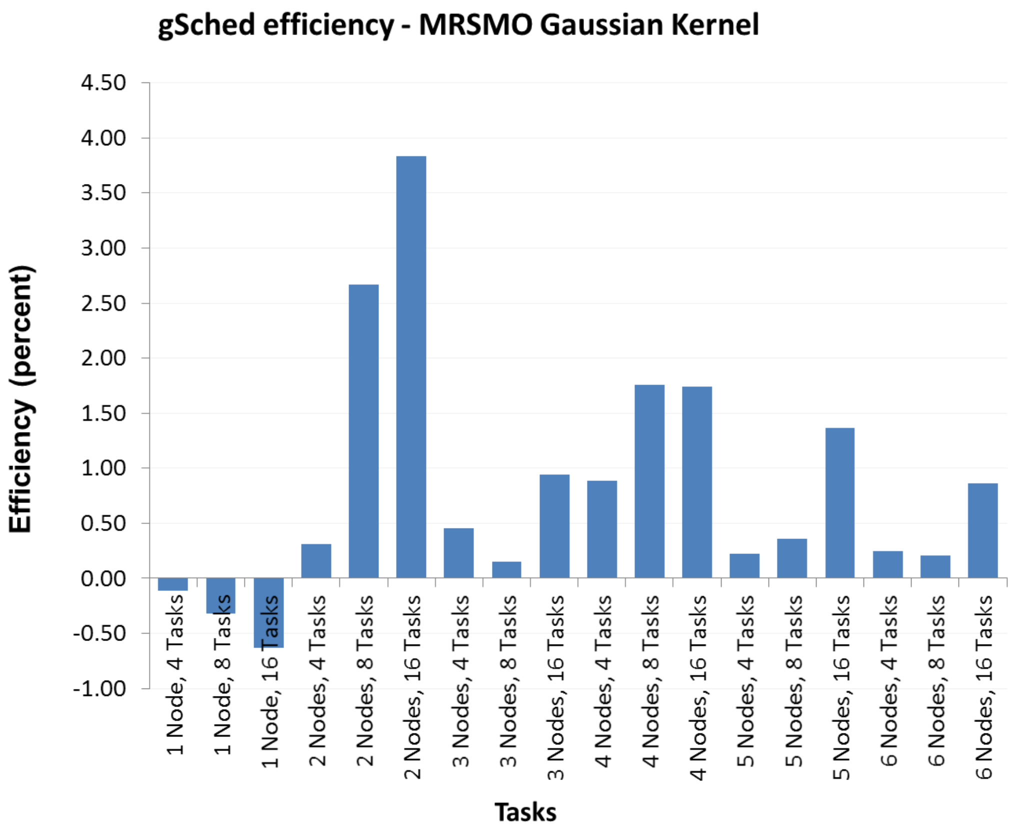


Figure 5-11: MRSMO - efficiency - Gaussian Kernel

gSched has a 'warm-up' period that is required to establish the underlying node characteristics as well as build the internal ECT table with information related to estimated compute times. The incremental machine learning classifier also needs to be trained with a contextually relevant number of instances (*which reflect task characteristics*) in order to be effective when trying to establish whether tasks are CPU or IO bound. During this time the matching and allocation scheme is ineffective - this until it has gained the required degree of insight to start effectively capitalizing on the proposed matching and allocation scheme. Furthermore, from Figure 5-8 and Figure 5-9 it is noted that gSched introduces an average of 0.5 seconds overhead for each participating node for the Polynomial kernel test (0.3 for the Guassian) when compared with the standard scheduler. From Figure 5-10 and Figure 5-11, we can also establish that gSched introduces an average penalty of 0.5 and 0.3 percent (Polynomial and Guassian tests) efficiency reduction for each node, again, when compared with the standard scheduler.

The average Polynomial kernel speedup, in seconds, of gSched over the standard scheduler ranges between a minimum of -1.07 and maximum of 12.10 times for the single and 4 node tests. The initial negative performance outcome is due to gSched's lack of task information. The equivalent figures for the Gaussian experiment range from -0.71 (*minimum*) to 13.94 (*maximum*) for the 1 node and 2 node experiments. In terms of efficiency, the experiment demonstrates a minimum of -0.53 and maximum of 1.51 percent improvement for the 1 and 5 node tests. The Gaussian experiment equivalents range from -0.36 (*minimum*) to 1.46 (*maximum*) for the 1 node and 4 node experiments.

5.5.3 gSched Scalability and Cost Effectiveness

gSched has a number of parameters which govern its behaviour, the more important of which are summarized in Table 5-4.

Table 5-4: gSched parameters

Param.	Description	Value
dt	Distance threshold between source and target vectors to establish task and node similarity	**0.99**
cpior	The ratio of CPU versus IO time	**0.10**
fts	Finished tasks search span window	**30**
mst	Number of Tasks before firing gSched	**2**
hbcr	Time span for scheduler configuration reload the respective configuration	**60**
hbcnfr	Time span before the machine learning scheme is re-loaded with new training instances	**1000**
mrno	Maximum number of times a Node can refuse to allocate a task	**1**
ects	The maximum size of the ECT table	**100**
rofs	Rank offset for ECT node selector	**1**

A further experiment with 10 mixed EC2 nodes was performed. This was intended to establish how the default gSched configuration performed when compared with the default Hadoop scheduler using the WordCount test [30]. The EC2 instance types employed for this experiment are described in Table 5-5:

Table 5-5: EC2 instance types - initial experiment

Type	Number	Hadoop Role
m1.medium	1	Master
c1.medium	1	Slave
t1.micro	5	Slave
t1.small	3	Slave

A number of concurrent jobs are executed, using the same input set. Figure 5-12 shows the performance of gSched (*and using the configuration shown in Table 5-5*) compared with the default Hadoop scheduler using the baseline Hadoop configuration in this scenario. Performance is similar up to 20 concurrent jobs. Subsequently however, gSched starts to outperform the default scheduler - established via the respective slope ($^{\Delta}y/^{\Delta}x$).

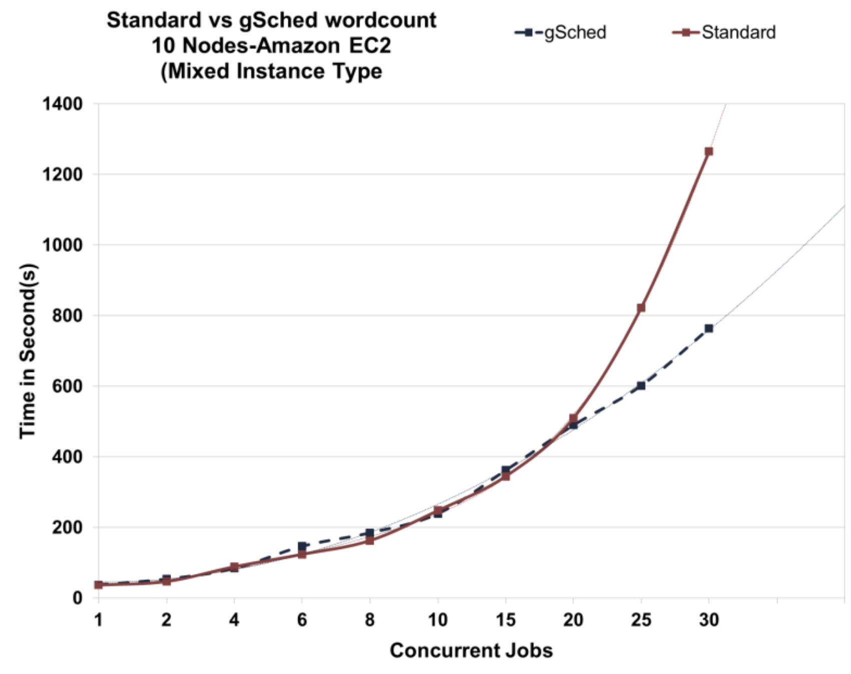
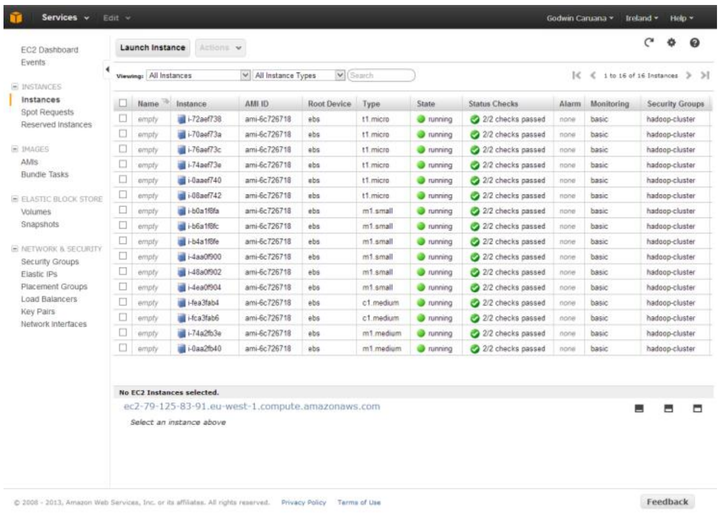


Figure 5-12: Word count Test - Standard vs. gSched

Given the associated costs in a cloud computing construct, which vary according to the respective node capabilities and use, the ability to maximize acquired resources capabilities is important. Reducing the number of speculative copies/failed tasks becomes an important perspective to control - these influence the respective processing and cost counters associated with the service.

Another experiment set was performed using 16 Amazon EC2 instances (Figure 5-13). Once again, the WordCount test is employed as a baseline.

Table 5-6 summarizes the cluster setup to this extent. Table 5-7 portrays 3 different sets of gSched configurations employed.



Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm	Monitoring	Security Groups
empty	i-72aef738	ami-6c726718	ebs	t1.micro	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-70aef73a	ami-6c726718	ebs	t1.micro	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-76aef73c	ami-6c726718	ebs	t1.micro	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-74aef73e	ami-6c726718	ebs	t1.micro	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-0aaef740	ami-6c726718	ebs	t1.micro	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-08aef742	ami-6c726718	ebs	t1.micro	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-b0a1f8fa	ami-6c726718	ebs	m1.small	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-b6a1f8fc	ami-6c726718	ebs	m1.small	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-b4a1f8fe	ami-6c726718	ebs	m1.small	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-4aa0f900	ami-6c726718	ebs	m1.small	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-48a0f902	ami-6c726718	ebs	m1.small	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-4ea0f904	ami-6c726718	ebs	m1.small	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-fea3fab4	ami-6c726718	ebs	c1.medium	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-fca3fab6	ami-6c726718	ebs	c1.medium	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-74a2fb3e	ami-6c726718	ebs	m1.medium	running	2/2 checks passed	none	basic	hadoop-cluster
empty	i-0aa2fb40	ami-6c726718	ebs	m1.medium	running	2/2 checks passed	none	basic	hadoop-cluster

No EC2 Instances selected.
ec2-79-125-83-91.eu-west-1.compute.amazonaws.com
Select an instance above

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Figure 5-13: Amazon EC2 Hadoop MapReduce Cluster based on specifically built AMI

Table 5-6: Amazon Instance Types

Type	Number	Hadoop Role
m1.medium	2	1 Master / 1 Slave
c1.medium	2	Slave
m1.small	6	Slave
t1.micro	6	Slave

Table 5-7:EC2 instance types - second experiment set

Experiment 1		Experiment 2		Experiment 3	
cpior	**0.10**	*cpior*	**0.12**	*cpior*	**0.10**
fts	30	*fts*	30	*fts*	30
mst	2	*mst*	2	*mst*	2
hbcr	60	*hbcr*	60	*hbcr*	60
hbcnfr	1000	*hbcnfr*	1200	*hbcnfr*	800
mrno	**2**	*mrno*	**1**	*mrno*	**2**
ects	100	*ects*	100	*ects*	100
rofs	**1**	*rofs*	**1**	*rofs*	**1**

Figure 5-14, Figure 5-15 and Figure 5-16 portray the performance of gSched when compared with the baseline Hadoop Scheduler using the respective configurations described in Table 5-7.

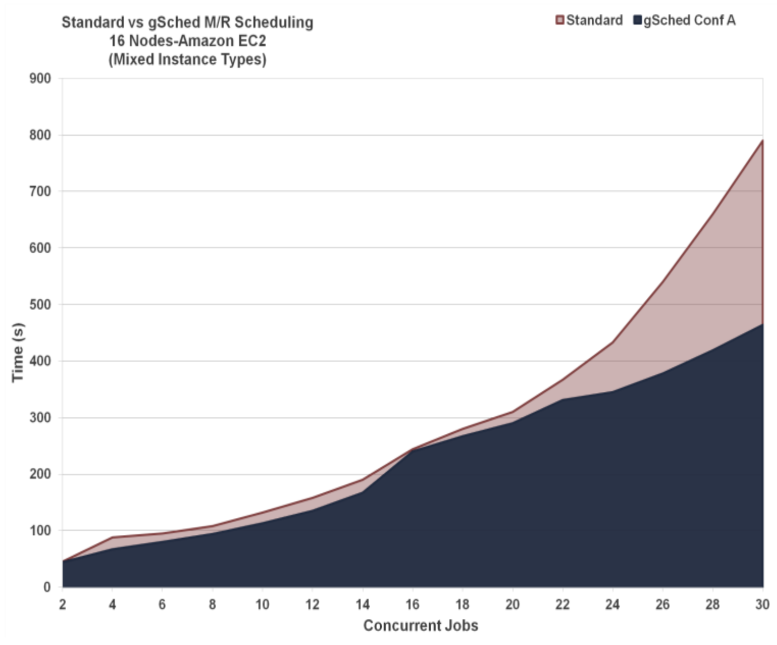


Figure 5-14: Cloud M/R - Standard vs. gSched (A)

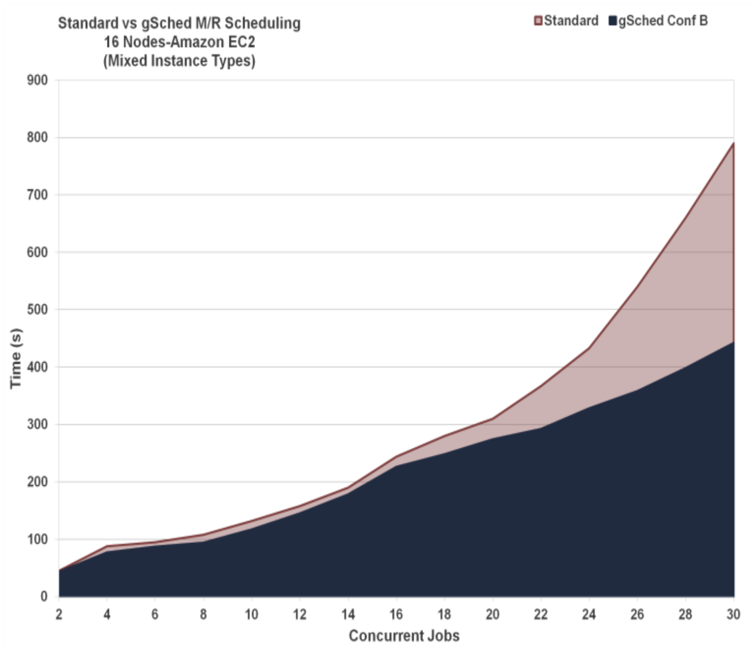


Figure 5-15: Cloud M/R - Standard vs. gSched (B)

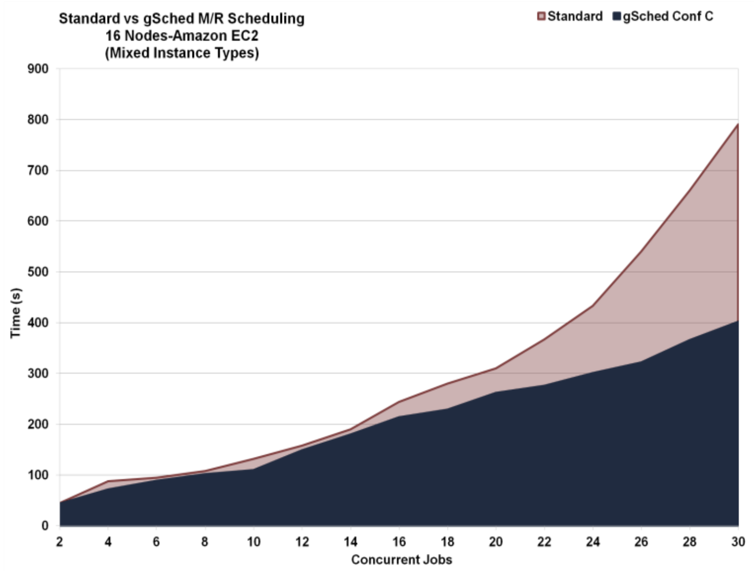


Figure 5-16: Cloud M/R - Standard vs. gSched (C)

More interestingly, Figure 5-17 shows the impact of speculative copies/straggler induced task errors exhibited via the default Hadoop and gSched (*using the three different configurations described*) scheduling approaches.

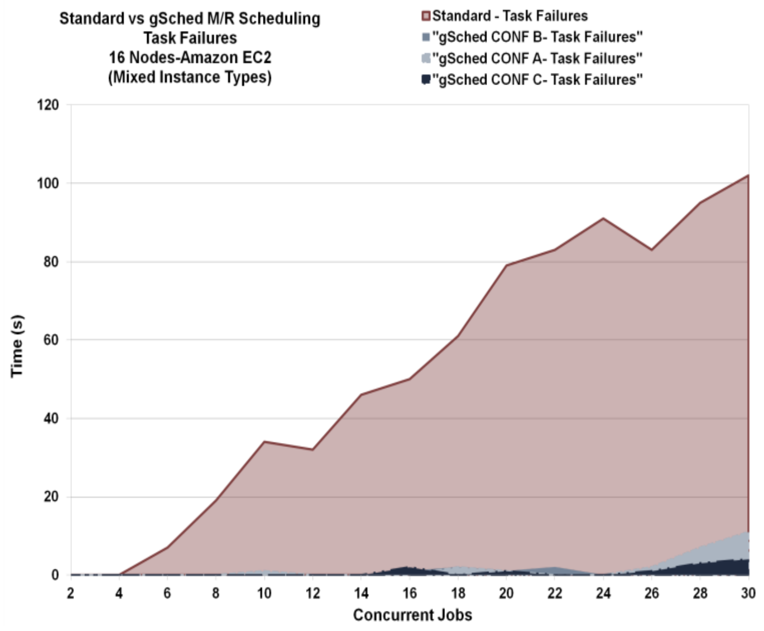


Figure 5-17: Cloud M/R - Standard vs. gSched Task Failures

Table 5-8 portrays the key performance differences between the three diverse gSched experimental configurations and the original scheduler in a Cloud context. The performance of the experiment is also believed to be similar with that of the (*and perhaps slightly better*

– *a direct comparison cannot be made explicitly by virtue of the rational presented earlier*) benchmark work presented in [239].

Table 5-8: Performance improvement

Experiment	Min	Median	Max
Standard vs. gSched (*Conf. A*)	1.64	14.40	**41.27**
Standard vs. gSched (*Conf. B*)	4.44	11.62	**44.05**
Standard vs. gSched (*Conf. C*)	2.22	16.67	**49.11**

A simple approximate area under trapezoid approach (*Newton-Cotes, Equation 5-10*) can be employed to establish the area differences under the curves of the graphs represented in Figure 5-14, Figure 5-15, Figure 5-16 and Figure 5-17 respectively.

The figures presented in Table 5-9 and Table 5-10 portray the improvement in terms of general 'cost' effectiveness and overall performance.

$$\int_{X0}^{X1} f(x)\,dx \approx h(f0+f1)/2 \qquad \textbf{[5-8]}$$

Table 5-9: gSched performance improvements over standard scheduler

Experiment	≈ Improvement
Standard vs. gSched (*Conf. A*)	**21%**
Standard vs. gSched (*Conf. B*)	**24%**
Standard vs. gSched (*Conf. C*)	**28%**

Table 5-10: gSched and Standard scheduler un-productive task error differences

Experiment	≈ Improvement
Standard vs. gSched (*Task Errors Conf. A*)	**97%**
Standard vs. gSched (*Task Errors Conf. B*)	**98%**
Standard vs. gSched (*Task Errors Conf. C*)	**99%**

The approach adopted by the default Hadoop scheduler is based on a progress rate of a task. In a highly multiplexed Cloud environment, this approach is not effective in the long run because the actual performance characteristics of participating nodes can change over time. This induces the default scheduler to launch un-productive speculative tasks ($r_{\text{unproductive}}$), beside resources which are minimally required ($r_{\text{productive}}$) which come at a price – in [239] the authors state that this can be as much as 80% of the number of tasks. In contrast with [239], gSched attempts to schedule tasks, including speculative ones, on those which are more reflective of the task characteristics.

From Equation 5-1, the charge rate difference between the standard and gSched allocation schemes, crd can be defined as:

$$crd = c(Standard) - c(gSched) \quad [5\text{-}9]$$

$$T_{gSched} = T_{standard - (P/_{100} * T_{standard})} \quad [5\text{-}10]$$

$$R_{gSched} = R_{standard - (E/_{100} * T_{standard})} \quad [5\text{-}11]$$

$$crd = \sum_{t=1}^{T_{standard}} \sum_{n=1}^{N_{standard}} R_{standard}(n) * u - \sum_{t=1}^{T_{gSched}} \sum_{n=1}^{N_{gSched}} R_{gSched}(n_{gSched}) * u \quad [5\text{-}12]$$

where:

- *E* is the sum of $r_{productive}$ and $r_{unproductive}$
- *P* is the performance improvement over the standard scheduler T

In a cost construct, and using a typical EC2 pay-per-use billing hour as an optimization constraint for the task allocation scheme, one can establish that non-productive speculative execution and copy errors, 'push' jobs into additional cost cycles (billing hours in the context of Amazon EC2). The proposed approach significantly reduces errors as shown in Table 5-10 whilst also improving on the performance of the original scheduler. This establishes that gSched meets the objectives laid out earlier (Equation 5-2).

Taking into consideration job concurrency as well as the number of errors the standard scheduler exhibits (*which impact R in Equation 5-2 and also translated into failed tasks and thus potentially re-scheduled, in turn increasing resource utilization time T*), this difference becomes increasingly more significant. Beyond the overall performance improvement, this is believed to have a positive contribution in terms of cost savings. This even without the explicit consideration of the cost of storage and network I/O [248][249]. Furthermore, the overall performance of gSched can be observed to be more “linear” in comparison to the original scheduler. On the basis of Equation 5-1, this is also construed as an important attribute for the ability to design MapReduce cluster requirements for scalable, more cost effective job processing.

5.6 Summary

Achieving a balance between performance and cost remains a challenge in highly multiplexed, heterogeneous cloud based scenarios [240], [250],[229], [249]. Executing a set of tasks {$t_1, t_2, t_3, \dots t_n$} on a set of machines with different capabilities {$m_1, m_2, m_3, \dots\dots m_n$} in a cost and performance effective way forms the basis of the work presented in this chapter.

Hadoop is a popular M/R implementation in cloud computing environments. Hadoop's default MapReduce scheduling approach is somewhat inefficient for heterogeneous environments. It assumes a degree of homogeneity which, in modern computing scenarios, especially those which are cloud based, is not common. Furthermore, for the latter, the underlying resource capabilities can actually vary during job execution. The differences and variation of these capabilities must be taken in consideration in order to optimize any scheduling strategy from a cost and performance perspective.

In this Chapter, gSched, a heterogeneous aware MapReduce scheduler implemented in a Hadoop context is presented. The proposed approach tries to achieve a balance between maximising resources, minimizing 'cost' and performance – a combination which remains a challenge in a highly multiplexed, heterogeneous cloud based scenario [240], [250],[229], [249]. The cost standpoint becomes increasingly relevant for task to node matching in such a context - a construct not researched extensively to date.

A specific Amazon EC2 image that can be employed as a baseline configuration for the provisioning of a MapReduce, spam filter training architecture in a cloud computing environment is also configured and deployed. A number of tests are performed, varying the task matching scheme's configuration to establish an optimal baseline to compare MRSMO's performance using the default and the proposed scheduling schemes. This both on an experimental, local, cluster as well as on an Amazon's EC2 cloud computing platform. The performance and efficiency characteristics between the two are identified, with gSched showing interesting improvements. The degree of maximisation of heterogeneity in the context of improved cost effectiveness is also explored and discussed, using the standard Hadoop WordCount test. The chapter concludes with a review of related work.

CHAPTER 6 – Conclusion and Future Work

This chapter summarises the key contributions as well as boundaries of this dissertation. It also articulates a succinct set of potential opportunities for improvement work and establishes the areas where this submission can be taken forward in terms of innovative research directions.

6.1 Conclusions

This research work investigated evolving approaches towards spam filtering. Numerous trends continue to emerge and evolve, including those based on peer to peer and grid computing, semantic and social and network based approaches amongst others. Whilst various schemes exist, machine learning based techniques have been applied extensively in spam filter training and classification constructs – also assisted with various techniques, including those based on RDF, for accuracy improvements. Combinations of approaches bring increased performance, accuracy and scalability opportunities. They also enable specific perspectives, including the formalization of spam filtering flow, definition, personalization as well as the ability to provide collaborative ecosystems towards improves spam filtering via sharing of computing resources and spam intelligence in general.

In real world contexts, most scale spam filtering implementations follow architectural styles and patterns that are mostly '*centralised*'. Actual implementations in this construct vary from on premise to off-site, service based approaches and variations in between. The use of distributed computing and parallel frameworks for large scale spam filter training in these scenarios is considered a novel research area. In this dissertation work, the MapReduce framework has been identified as an enabling technology for high performance machine learning approaches to spam filter training. This rationale is considered as a key baseline for the research work performed towards establishing, presenting and evaluating, with success, a flexible, collaborative, distributed spam filter training architectural pattern. This architecture is designed to be able to scale at the rate mandated by the continuously evolving volume, velocity and variety of spam.

Rather than focusing on classification schemes or underlying architectures in isolation, a comprehensive level of consideration is applied to key dimensions concurrently, namely the underlying enabling infrastructure, the filter training scheme(s) as well as end user contribution and influence. The collaborative notion in this context can also extend beyond end users and include service providers, private and public sector institutions for example – this enabled by the representation of spam 'intelligence' with RDF based techniques. A long term objective is the suggestion and subsequent realization of an institutionalized global virtual organization supporting a global spam filtering 'ecosystem'.

This research work contributes MRSMO, a distributed, MapReduce based support vector machine algorithm for scalable and high performance spam filter training. The performance improvements when compared to the traditional sequential counterpart are apparent, with a 4x improvement recorded. On the other hand, the splitting of the training data for distributed, individual SVM solver computation and aggregation as presented introduced a deficiency in the accuracy of the classifier. To mitigate this discrepancy, this work proposes and evaluates an end user enabled, accuracy improving feedback process. This is achieved by improving and augmenting the base training data set(s) with additional intelligence sought from end users via an RDF based feedback loop. The training and testing sets are transformed to a Resource Description Framework (*RDF*) graph representation and misclassified instances identified using SPARQL [216]. The supplemented intelligence instance data is then re-deployed to the original training data sets and the MapReduce based MRSMO model is re-computed. Whilst SVM training is a compute intensive technique, the distributed MRSMO algorithm designed and the MapReduce framework allows for fast retraining of the model - something not feasible in a reasonably 'timely' fashion with traditional SVM approaches. The feedback loop, which can be performed any number of times and the learning model efficiently re-computed using MRSMO, improves the overall accuracy to and beyond the original Sequential SMO accuracy.

In order to further improve the overall approach and with the intent to come up with a 'turnkey' spam filtering stack which can be deployed in Internet scale scenarios, an improved task assignment method for MapReduce, using Hadoop and Amazon as enablers is proposed. Designated gSched, the contributed approach in this respect provides a heterogeneous aware, task to node allocation scheme, implemented in a Hadoop M/R construct. gSched tries to achieve a balance between maximising resources, minimizing 'cost' and performance (*a combination which remains a challenge in highly multiplexed, heterogeneous cloud based scenarios where the underlying resources performance can change significantly over their lifespan*). This also based on the specially configured and deployed Amazon EC2 image which can be employed as a baseline configuration for the provisioning of a holistic MapReduce, spam filter training architecture in a cloud computing environment. The tests performed, varying the task matching scheme's configuration to establish the performance and efficiency characteristics required, show the performance differences between (*and improvements over*) the baseline Hadoop scheduler and the proposed approach.

The combination of the RDF based feedback loop, the M/R based distributed support vector machine and the task to node matching scheme designed and evaluated are believed to contribute a novel holistic approach towards a high performance, accurate and commodity based spam filter training architectural pattern – realizing the original objectives of this research work.

6.2 Future Work

The application and exploit of ontology based techniques has considerable potential in the context of spam filtering and classification beyond what has been explored in this research work. The design of a fully-fledged, yet abstract spam ontology in conjunction with a domain specific reasoner can be designed and developed to increase the value proposition of the current prototype. Currently, the RDF based feedback loop approach is based and capitalizes on human expertise to identify additional context. Thus, research into enhanced approaches how to automatically extract additional, hidden and latent intelligence from training instances, subsequently employed with the feedback loop is considered an improvement opportunity.

Latent Semantic Analysis (LSA) and Singular Value Decomposition (SVD), for example, are techniques which can be employed during various stages for accuracy improvement in a specific spam filter training construct. The respective consideration and application within the distributed SVM process or separately parallelized in a similar fashion can exploit the performance and scalability characteristics of the M/R framework [185]. From a high level perspective, from Algorithm 3-2 (Pg. 49), the 'raw' input instance vectors (data set A – an $m \times n$ matrix) of data-set fragments on each node are transformed via a singular value decomposing process, described more formally as $A = U \Sigma V^T$, where $rank(A) = r$, U is the word vector and V is the document vector.

Further research opportunities surrounding the optimization of the Hadoop MapReduce model in a heterogeneous computing construct - from a performance, financial as well as benefit realization perspective is also believed appropriate. Limited work which specifically focuses on optimizing, comparing and establishing baselines between performance, cost modelling and chargeback approaches [8][6] has been identified during this research. The same applies in terms of the opportunity to improve gSched's consideration of the underlying network capabilities, data placement and data 'sharding' – how to best split the data in of MapReduce task at hand and cluster attributes, job concurrency and operating system scheduler conflict (*including host and hypervisor context switching*). As presented, gSched is not able to establish and exploit the underlying network performance characteristics nor employs targeted strategies in terms of data splitting for node distribution. Furthermore, the current Bayesian classification for establishing task bias should be abstracted and packaged in a plug-in based construct so it can be replaced by schemes which can improved the overall efficiency of the task to node allocation scheme in specific scenarios (*such as Genetic Algorithms etc.*). gSched's flexibility could also be further enhanced by the introduction of additional scheduling constraints, including priority awareness and processing 'budget' for example.

Resource profiling and benchmarking has been discovered to be non-straight forward in a highly multiplexed, virtual resource provisioning construct [251]. This impacts the underlying performance, especially from a consistency and behaviour perspective, considerably. Thus, for future work, it is intended that larger scale tests in terms of instances and workloads are performed, which will allow to increasingly fine tune and improve gSched. It will also allow the introduction of additional features that are able to influence the runtime behaviour of the scheduler depending on the context.

The longer term objective also includes the ability to automatically configure the MapReduce cluster towards specific performance and costs objectives. This needs to take into consideration the underlying provisioning stack characteristics, the multiplexing scheduling schemes of the underlying hyper visor/host as well as the MapReduce guest operating environments.

The entire value proposition can be further strengthened considerably with the *domain specific language* which can be used to:

1. Define the behaviour and parameters of a distributed machine learning scheme (in the proposed case, the SVM – MRSMO).
2. Define and enable the automated inference and correlation behaviour of the RDF based feedback loop. This perspective should also include the required degree of consideration for assistive techniques such latent semantic indexing and analysis.
3. Describe the MapReduce cluster characteristics for automated cluster creation and start-up. This can include the number of instances, time to live, cost capping etc.
4. Describe and govern the runtime behaviour of the task allocation scheme (in this case gSched).

The principled motive and objective continues to be the research and study of high performance, accurate, commodity based spam filtering architectures - from a holistic perspective.

References

[1] L. Gomes, C. Cazita, and J. Almeida, “Characterizing a spam traffic,” *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pp. 356–369, 2004.

[2] P. P. Paul, P. Judge, D. Alperovitch, and W. Yang, “Understanding and Reversing the Profit Model of Spam,” in *4th Annual Workshop on the Economics of Information Security, WEIS 2005, Harvard University*, 2005.

[3] W. Gansterer and M. Ilger, “Anti-spam methods—state of the art,” *Technical Report FA384018-1, Institute of Distributed and Multimedia Systems, University of Vienna*, 2005.

[4] E. Blanzieri and A. Bryl, “A survey of learning-based techniques of email spam filtering,” *Artif. Intell. Rev.*, vol. 29, no. 1, pp. 63–92, 2008.

[5] J. R. Levine, “Experiences with greylisting,” in *Second Conference on Email and Anti-Spam*, 2005.

[6] J. Carpinter and R. Hunt, “Tightening the net: A review of current and next generation spam filtering tools,” *Computers & Security*, vol. 25, no. 8, pp. 566 – 78, 2006.

[7] G. Schryen, “Armed for the Spam Battle: A Technological and Organizational Infrastructure Framework,” in *Hawaii International Conference on System Sciences*, p. 144, 2007.

[8] A. Conry-Murray, “The antispam cocktail: mix it up to stop junk E-mail,” *Network Magazine*, vol. 19, no. 7, pp. 33 – 8, 2004.

[9] Á. Blanco, A. Ricket, and M. Martín-Merino, “Combining SVM classifiers for email anti-spam filtering,” in *IWANN’07 Proceedings of the 9th international work conference on Artificial neural networks*, pp. 903 – 10, 2007.

[10] L. Mei and Y. Cheng-qing, “Anti-spam technique with SVM and K-NN cooperation method,” *Computer Engineering and Applications*, vol. 44, no. 4, pp. 135 – 7, Feb. 2008.

[11] J. C. Platt, *Advances in kernel methods*. Cambridge, MA, USA: MIT Press, pp. 185–208, 1999.

[12] M. Ye, T. Tao, F.-J. Mai, and X.-H. Cheng, “A Spam Discrimination Based on Mail Header Feature and SVM,” in *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–4, 2008.

[13] C. Xiao-li, L. Pei-yu, Z. Zhen-fang, and Q. Ye, “A method of spam filtering based on weighted support vector machines,” in *IT in Medicine & Education, 2009. ITIME ’09. IEEE International Symposium on*, vol. 1, pp. 947 –950, 2009.

[14] F. Colas and P. Brazdil, “Comparison of SVM and some Older Classification Algorithms in Text Classification Tasks,” in *Artificial Intelligence in Theory and Practice*, pp. 169–178, 2006.

[15] N. Cristianini and J. Shaw-Taylor, *An Introduction to SVM’s and Other Kernel-based Learning Methods.* New York, NY, USA: Cambridge University Press, 2000.

[16] T. Do, V. Nguyen, and F. Poulet, “Speed up SVM algorithm for massive classification tasks,” *Advanced Data Mining and Applications*, pp. 147–157, 2008.

[17] K.-L. K. Li, H.-K. Huang, S.-F. Tian, L. Kun-Lun, L. Kai, H. Hou-Kuan, and T. Sheng-Feng, “Active learning with simplified SVMs for spam categorization,” in *Proceedings. International Conference on Machine Learning and Cybernetics*, vol. 3, pp. 1198–1202, 2002.

[18] L. J. Cao, S. S. Keerthi, C.-J. J. Ong, J. Q. Zhang, U. Periyathamby, X. J. J. Fu, and H. P. Lee, “Parallel sequential minimal optimization for the training of support vector machines.,” *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 1039–1049, Jul. 2006.

[19] B. Catanzaro, N. Sundaram, and K. Keutzer, “Fast support vector machine training and classification on graphics processors,” in *Proceedings of the 25th international conference on Machine learning - ICML ’08*, pp. 104–111, 2008.

[20] H. Drucker, S. Member, D. Wu, S. Member, and V. N. Vapnik, “Support vector machines for spam categorization,” *IEEE Transactions on Neural Networks*, vol. 10, pp. 1048–1054, 1999.

[21] D. Sculley and G. M. Wachman, “Relaxed online SVMs for spam filtering,” in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR ’07*, p. 415, 2007.

[22] 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.

[23] S. Hershkop and S. J. Stolfo, “Identifying spam without peeking at the contents,” *Crossroads*, vol. 11, no. 2, pp. 3–3, 2004.

[24] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[25] F. R. Bach and M. I. Jordan, “Predictive low-rank decomposition for kernel methods,” in *Proceedings of the 22nd international conference*, pp. 33–40, 2005.

[26] G. Zanghirati and L. Zanni, “A parallel solver for large quadratic programs in training support vector machines,” *Parallel Computing*, vol. 29, no. 4, pp. 535–551, Apr. 2003.

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

[28] B. He, W. Fang, Q. Luo, N. K. Govindaraju, and T. Wang, “Mars: a MapReduce framework on graphics processors,” in *Proceedings of the 17th international conference on Parallel architectures and compilation techniques - PACT ’08*, p. 260, 2008.

[29] K. Taura, T. Endo, K. Kaneda, and A. Yonezawa, “Phoenix: a parallel programming model for accommodating dynamically joining/leaving resources,” in *SIGPLAN Not.*, vol. 38, no. 10, pp. 216–229, 2003.

[30] T. White, *Hadoop: The Definitive Guide, 1st Edition*, vol. 1. O’Reilly Media, Inc, 2009.

[31] T. Aarnio, “Parallel Data Processing with Mapreduce,” *TKK T-110.5190, Seminar on Internetworking*, 2009. [Online]. Available: http://www.cse.tkk.fi/en/publications/B/5/papers/Aarnio_final.pdf. [Accessed: 29-Apr-2010].

[32] A. S. Das, M. Datar, A. Garg, and S. Rajaram, “Google news personalization,” in *Proceedings of the 16th international conference on World Wide Web - WWW ’07*, p. 271, 2007.

[33] J. Dean, “Experiences with MapReduce, an abstraction for large-scale computation,” *PACT ’06 Proceedings of the 15th international conference on Parallel architectures and compilation techniques*, pp. 1–1, 2006.

[34] M. G. Noll and C. Meinel, “Building a Scalable Collaborative Web Filter with Free and Open Source Software,” in *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, pp. 563–571, 2008.

[35] “Forrester Research-Sizing The Cloud.” [Online]. Available: http://www.forrester.com/Sizing+The+Cloud/fulltext/-/E-RES58161?objectid=RES58161. [Accessed: 14-Dec-2012].

[36] “What is AWS?” [Online]. Available: http://aws.amazon.com/what-is-aws/. [Accessed: 14-Dec-2012].

[37] “Features: An Introduction to Windows Azure.” [Online]. Available: http://www.windowsazure.com/en-us/home/features/overview/. [Accessed: 14-Dec-2012].

[38] "Google Cloud Platform." [Online]. Available: https://cloud.google.com/. [Accessed: 14-Dec-2012].

[39] J. Strebel, "Cost Optimization Model for Business Applications in Virtualized Grid Environments," in *Grid Economics and Business Models*, no. 5745, pp. 74–87, 2009.

[40] J. Strebel and A. Stage, "An economic decision model for business software application deployment on hybrid Cloud environments.," in *Multikonferenz Wirtschaftsinformatik 2010*, pp. 195–206, 2010.

[41] B. Martens, M. Walterbusch, and F. Teuteberg, "Costing of Cloud Computing Services: A Total Cost of Ownership Approach," *Hawaii International Conference on System Sciences*, vol. 0, pp. 1563–1572, 2012.

[42] A. Wieder, P. Bhatotia, A. Post, and R. Rodrigues, "Brief announcement: modelling MapReduce for optimal execution in the cloud," in *Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pp. 408–409, 2010.

[43] T. S. Guzella and W. M. Caminhas, "A review of machine learning approaches to Spam filtering," *Expert Syst. Appl.*, vol. 36, no. 7, pp. 10206–10222, 2009.

[44] R. Schollmeier, "A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications," in *P2P '01 Proceedings of the First International Conference on Peer-to-Peer Computing*, p. 101, 2001.

[45] M. Li and M. Baker, *The grid core technologies*. John Wiley & Sons, 2005.

[46] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, no. 5, pp. 34–43, 2001.

[47] D. Boyd and N. B. Ellison, "Social Network Sites: Definition, History, and Scholarship," *Journal of Computer-Mediated Communication*, vol. 13, no. 1–2, Nov. 2007.

[48] R. Roman, J. Zhou, and J. Lopez, "An anti-spam scheme using pre-challenges," *Computer Communications*, vol. 29, no. 15, pp. 2739–2749, 2006.

[49] Z. Duan, Y. Dong, and K. Gopalan, “DMTP: Controlling spam through message delivery differentiation,” *Comput. Netw.*, vol. 51, no. 10, pp. 2616–2630, 2007.

[50] Q. Li and B. Mu, “A Novel Method to Detect Junk Mail Traffic,” in *2009 Ninth International Conference on Hybrid Intelligent Systems*, pp. 129–133, 2009.

[51] D. Twining, M. M. Williamson, M. J. F. Mowbray, and M. Rahmouni, “Email prioritization: reducing delays on legitimate mail caused by junk mail,” in *ATEC ’04 Proceedings of the annual conference on USENIX Annual Technical Conference*, p. 4, 2004.

[52] W. Gansterer and M. Ilger, “Analyzing uce/ube traffic,” *ICEC ’07 Proceedings of the ninth international conference on Electronic commerce*, pp. 195–204, 2007.

[53] SpamHaus, “SpamHaus,” 1998. [Online]. Available: http://www.spamhaus.org/. [Accessed: 02-Apr-2010].

[54] SpamCop, “CISCO Systems.” [Online]. Available: http://www.spamcop.net. [Accessed: 03-Jul-2010].

[55] DNSBL, “Domain Name System Blacklists.” [Online]. Available: http://www.dnsbl.info/. [Accessed: 23-Jun-2010].

[56] A. Ramachandran, N. Feamster, and S. Vempala, “Filtering spam with behavioral blacklisting,” in *CCS ’07 Proceedings of the 14th ACM conference on Computer and communications security*, pp. 342–351, 2007.

[57] J. Goodman and R. Rounthwaite, “Stopping outgoing spam,” in *EC ’04: Proceedings of the 5th ACM conference on Electronic commerce*, pp. 30–39, 2004.

[58] D. Cook and J. Hartnett, “Catching spam before it arrives: domain specific dynamic blacklists,” in *ACSW Frontiers ’06 Proceedings of the 2006 Australasian workshops on Grid computing and e-research - Volume 54*, pp. 193–202, 2006.

[59] F. Sebastiani and C. N. D. Ricerche, “Machine Learning in Automated Text Categorization,” *ACM Computing Surveys*, vol. 34, pp. 1–47, 2002.

[60] Y. Yang and J. O. Pedersen, “A Comparative Study on Feature Selection in Text Categorization,” in *ICML ’97 Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 412–420, 1997.

[61] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Information processing & management*, pp. 513–523, 1988.

[62] J. G. Hidalgo, M. López, and E. Sanz, “Combining text and heuristics for cost-sensitive spam filtering,” in *ConLL ’00 Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning - Volume 7*, pp. 99–102, 2000.

[63] P. Gburzynski and J. Maitan, “Fighting the spam wars: A remailer approach with restrictive aliasing,” *ACM Trans. Internet Technol.*, vol. 4, no. 1, pp. 1–30, 2004.

[64] A. G. K. Janecek, W. N. Gansterer, and K. A. Kumar, “Multi-Level Reputation-Based Greylisting,” in *2008 Third International Conference on Availability, Reliability and Security*, pp. 10–17, 2008.

[65] A. Khanal, B. Motlagh, and T. Kocak, “Improving the Efficiency of Spam Filtering through Cache Architecture,” in *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007. MASCOTS ’07. 15th International Symposium on*, pp. 303–309, 2007.

[66] S. Yih, R. McCann, and A. Kolcz, “Improve Spam Filtering by Detecting Gray Mail,” in *Proceedings of the 4th Conference on Email and Anti-Spam*, 2007.

[67] G. V Cormack and T. R. Lynam, “Online supervised spam filter evaluation,” *ACM Trans. Inf. Syst.*, vol. 25, no. 3, p. 11, 2007.

[68] P. Hayati and V. Potdar, “Evaluation of spam detection and prevention frameworks for email and image spam: a state of art,” in *iiWAS ’08 Proceedings of the 10th*

International Conference on Information Integration and Web-based Applications & Services, pp. 520–527, 2008.

[69] S. Hershkop and S. Stolfo, “Combining email models for false positive reduction,” in *KDD ’05 Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pp. 98–107, 2005.

[70] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar, “Can machine learning be secure?,” in *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pp. 16–25, 2006.

[71] G. M. Weiss and Y. Tian, “Maximizing classifier utility when training data is costly,” *SIGKDD Explor. Newsl.*, vol. 8, no. 2, pp. 31–38, Dec. 2006.

[72] Limewire, “The Gnutella Protocol Specification,” *The Gnutella Protocol Specification*, 2001. [Online]. Available: http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.

[73] “Limewire,” *Peer To Peer & Torrent Client*, 2009. [Online]. Available: http://www.limewire.com.

[74] A. Rowstron and P. Druschel, “Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems,” in *Middleware ’01 Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, pp. 329–350, 2001.

[75] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications,” in *SIGCOMM ’01 Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 149–160, 2001.

[76] N. Dimmock and I. Maddison, “Peer-to-peer collaborative spam detection,” *Crossroads*, vol. 11, no. 2, pp. 4–4, 2004.

[77] A. Gray and M. Haahr, “Personalised, Collaborative Spam Filtering,” in *CEAS Proceedings of the first conference on Email and Anti-Spam*, 2004.

[78] P. Luo, H. Xiong, K. Lü, and Z. Shi, “Distributed classification in peer-to-peer networks,” in *KDD ’07 Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 968–976, 2007.

[79] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati, “P2P-based collaborative spam detection and filtering,” in *P2P ’04 Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, pp. 176 – 83, 2004.

[80] F. Zhou, L. Zhuang, B. Y. Zhao, L. Huang, A. D. Joseph, and J. Kubiatowicz, “Approximate object location and spam filtering on peer-to-peer systems,” in *Middleware ’03 Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware*, pp. 1–20, 2003.

[81] J. Metzger, M. Schillo, and K. Fischer, “A Multiagent-Based Peer-to-Peer Network in Java for Distributed Spam Filtering.” [Online]. Available: http://www.springerlink.com/content/0t3wgt1auxk2xqaa/. [Accessed: 05-Jul-2009].

[82] N. Foukia, L. Zhou, and C. Neuman, “Multilateral decisions for collaborative defense against unsolicited bulk e-mail,” *Trust Management*, pp. 77–92, 2006.

[83] J. S. Kong, B. A. Rezaei, N. Sarshar, V. P. Roychowdhury, and P. O. Boykin, “Collaborative spam filtering using e-mail networks,” *COMPUTER*, vol. 39, no. 8, p. 67+, Aug. 2006.

[84] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach*, 5th ed. USA: Addison-Wesley Publishing Company, 2009.

[85] Microsoft, “Hosted Exchange Services,” *Microsoft Online Services*, 2010. [Online]. Available: http://www.microsoft.com/online/exchange-hosted-services.mspx .

[86] GFI, “Hosted Spam Filtering Service,” 2010. [Online]. Available: http://www.gfi.com/landing/maxmp-hosted-spam-filtering-service.asp?adv=69&loc=643&gclid=CJCy9bqVuKACFUmS3wodhFQaTQ.

[87] P. Liu, G. Chen, L. Ye, and W. Zhong, “Anti-spam grid: a dynamically organized spam filtering infrastructure,” in *SMO’05 Proceedings of the 5th WSEAS international conference on Simulation, modelling and optimization*, pp. 67–72, 2005.

[88] Z. Yuan, Q. Zhang, D. Li, and Y. Liu, “Research of Anti-spam Application Architecture Based on Semantic Grid,” in *ICICIC ’07 Proceedings of the Second International Conference on Innovative Computing, Informatio and Control*, p. 491, 2007.

[89] H. Ohfuku and K. MATSUURA, “Integration of Bayesian filter and social network technique for combating spam e-mails better,” *Trans. Info. Process. Soc. Japan*, vol. 47, no. 8, pp. 2548 – 55, 2006.

[90] P. O. Boykin and V. P. Roychowdhury, “Leveraging Social Networks to Fight Spam,” *Computer*, vol. 38, no. 4, pp. 61–68, 2005.

[91] P.-A. Chirita, J. Diederich, and W. Nejdl, “MailRank: using ranking for spam detection,” in *CIKM ’05 Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 373–380, 2005.

[92] S. Brin and L. Page, “The anatomy of a large-scale hypertextual Web search engine,” *Comput. Netw. ISDN Syst.*, vol. 30, no. 1–7, pp. 107–117, 1998.

[93] A. Garg, R. Battiti, and R. G. Cascella, “‘May I borrow your filter?’ Exchanging filters to combat spam in a community,” in *AINA ’06 Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 02*, vol. 2, p. 5 pp.–, 2006.

[94] V. Potdar, F. Ridzuan, and J. Singh, “Spam 2.0,” in *Proceedings of the CUBE International Information Technology Conference on - CUBE ’12*, p. 724, 2012.

[95] A. Zinman and J. S. Donath, “Is Britney Spears Spam?,” in *CEAS*, 2007.

[96] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, C. Zhang, and K. Ross, "Identifying video spammers in online social networks," in *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, pp. 45–52, 2008.

[97] G. Brown, T. Howe, M. Ihbe, A. Prakash, and K. Borders, "Social networks and context-aware spam," in *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pp. 403–412, 2008.

[98] C.-F. Yeh, C.-H. Mao, H.-M. Lee, and T. Chen, "Adaptive e-mail intention finding mechanism based on e-mail words social networks," in *Proceedings of the 2007 workshop on Large scale attack defense*, pp. 113–120, 2007.

[99] S. Youn and D. McLeod, "Improved spam filtering by extraction of information from text embedded image e-mail," in *Proceedings of the 2009 ACM symposium on Applied Computing - SAC '09*, p. 1754, 2009.

[100] J. Kim, D. Dou, H. Liu, and D. Kwak, "Constructing a User Preference Ontology for Anti-spam Mail Systems," in *CAI '07 Proceedings of the 20th conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence*, pp. 272–283, 2007.

[101] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, p. 10, Nov. 2009.

[102] M. Balakumar and V. Vaidehi, "Ontology based classification and categorization of email," in *2008 International Conference on Signal Processing, Communications and Networking*, pp. 199–202, 2008.

[103] S. Youn and D. McLeod, "Spam decisions on gray e-mail using personalized ontologies," in *SAC '09 Proceedings of the 2009 ACM symposium on Applied Computing*, pp. 1262–1266, 2009.

[104] S. Youn and D. McLeod, “Efficient spam email filtering using adaptive ontology,” in *Fourth International Conference on Information Technology ITNG07*, pp. 249–254, 2007.

[105] J.-H. Hsia and M.-S. Chen, “Language-model-based detection cascade for efficient classification of image-based spam e-mail,” in *ICME’09 Proceedings of the 2009 IEEE international conference on Multimedia and Expo*, pp. 1182–1185, 2009.

[106] B. Orgun, M. Dras, A. Nayak, and G. James, “Approaches for semantic interoperability between domain ontologies,” *Expert Systems*, pp. 41–50, 2008.

[107] A. Ramachandran and N. Feamster, “Understanding the network-level behavior of spammers,” in *SIGCOMM ’06 Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 291–302, 2006.

[108] A. Pathak, Y. C. Hu, and Z. M. Mao, “Peeking into spammer behavior from a unique vantage point,” in *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, pp. 3:1–3:9, 2008.

[109] P. H. C. Guerra, D. Guedes, J. Wagner Meira, C. Hoepers, M. H. P. C. Chaves, and K. Steding-Jessen, “Spamming Chains: A New Way of Understanding Spammer Behavior,” in *Proceedings of the 6th Conference on e-mail and anti-spam (CEAS)*, 2009.

[110] A. A. Antonopoulos, K. G. Stefanidis, and A. G. Voyiatzis, “Fighting spammers with spam,” in *Autonomous Decentralized Systems, 2009. ISADS ’09. International Symposium on*, pp. 1–5, 2009.

[111] Y. Shinjo, K. Matsui, T. Sugimoto, and A. Sato, “An anti-spam scheme using capability-based access control,” in *IEEE 34th Conference on Local Computer Networks, 2009. LCN 2009.*, pp. 907–914, 2009.

[112] H. Esquivel, T. Mori, and A. Akella, “Router-Level Spam Filtering Using TCP Fingerprints: Architecture and Measurement-Based Evaluation.” .

[113] A. Brodsky and D. Brodsky, “Trinitya: A distributed defense against transient spam-bots,” in *PODC ’07 Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*, pp. 378–379, 2007.

[114] T. A. S. Project, “Spamassasin,” 2010. [Online]. Available: http://spamassassin.apache.org/.

[115] Y. Xie, F. Yu, K. Achan, and R. Panigrahy, “Spamming botnets: signatures and characteristics,” in *SIGCOMM ’08 Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pp. 171–182, 2008.

[116] C. Wei, A. Sprague, and G. Warner, “Clustering malware-generated spam emails with a novel fuzzy string matching algorithm,” in *SAC ’09 Proceedings of the 2009 ACM symposium on Applied Computing*, pp. 889–890, 2009.

[117] E. Allman, “E-mail Authentication: What, Why, How?,” *Queue*, vol. 4, no. 9, pp. 30–34, 2006.

[118] G. Lawton, “E-mail authentication is here, but has it arrived yet?,” *Computer*, vol. 38, no. 11, pp. 17–19, 2005.

[119] P. Peterson, “E-mail is broke! Authentication can fix it,” *Electronic Design*, vol. 54, no. 11, p. 22, 2006.

[120] L. Liao and J. Schwenk, “End-to-end header protection in signed S/MIME,” *OTM’07 Proceedings of the 2007 OTM confederated international conference on On the move to meaningful internet systems: CoopIS, DOA, ODBASE, GADA, and IS*, vol. Volume Par, pp. 1646–1658, 2007.

[121] “Spammers embrace email authentication • The Register.” [Online]. Available: http://www.theregister.co.uk/2004/09/03/email_authentication_spam/. [Accessed: 14-Aug-2009].

[122] P. Klangpraphant and P. Bhattarakosol, “e-mail authentication system: a spam filtering for smart senders,” *ICIS ’09 Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, pp. 534–538, 2009.

[123] R. Wetzel, “Spam fighting business models-who wins, who loses,” *Business Communications Review*, vol. 34, no. 4, pp. 24 – 9, 2004.

[124] L. von Ahn, M. Blum, and J. Langford, “Telling humans and computers apart automatically,” *Commun. ACM*, vol. 47, no. 2, pp. 56–60, 2004.

[125] S. Shirali-Shahreza and A. Movaghar, “A new anti-spam protocol using CAPTCHA,” in *Proceedings of the IEEE International Conference on Networking, Sensing and Control, ICNSC 2007*, pp. 234–238, 2007.

[126] P. Lin, J. Huang, and T. Chang, “CAPTCHA-based anti-spam mail model,” in *Proceedings of the International Conference on Security and Management, SAM ’04*, pp. 332–338, 2004.

[127] IETF, “SASL,” *RFC 2222*, 2010. [Online]. Available: http://www.ietf.org/rfc/rfc2222.txt.

[128] K. Curran and J. Honan, “Addressing spam e-mail using hashcast,” *Journal of Business Data Communications and Networking*, vol. 1, no. 2, pp. 41–65, Jul. 2005.

[129] K. Li and Z. Zhong, “Fast statistical spam filter by approximate classifications,” *ACM SIGMETRICS Performance Evaluation Review*, pp. 347–358, 2006.

[130] N. Denny, T. El Hourani, J. Denny, S. Bissmeyer, and D. Irby, “SpamCooker: A Method for Deterring Unsolicited Electronic Communications,” in *Third International Conference on Information Technology: New Generations (ITNG’06)*, pp. 590–591, 2006.

[131] J. Bentley and C. Mallows, “CAPTCHA challenge strings: Problems and improvements,” *Electronic Imaging 2006*, vol. 6067, pp. 60670 – 1, 2006.

[132] J. Yan and A. El Ahmad, “A Low-cost Attack on a Microsoft CAPTCHA,” in *CCS ’08 Proceedings of the 15th ACM conference on Computer and communications security*, pp. 543–554, 2008.

[133] W. Gansterer, H. Hlavacs, and M. Ilger, “Token buckets for outgoing spam prevention,” in *Proceedings of IASTED International Conference on Communication, Network, and Information Security*, pp. 36 – 41, 2005.

[134] D. A. Turner and D. M. Havey, “Controlling Spam through Lightweight Currency,” in *Proceedings Of The Hawaii International Conference On Computer Sciences*, 2004.

[135] S. Radicati and M. Khmartseva, “Email Statistics Report.” The Radicati Group, 2009.

[136] C. Lueg and S. Martin, “Users dealing with spam and spam filters: some observations and recommendations,” *Proceedings of the 8th ACM SIGCHI New Zealand …*, pp. 67–72, 2007.

[137] K. Junejo and A. Karim, “PSSF: A novel statistical approach for personalized service-side spam filtering,” *WI ’07 Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 228–234, 2007.

[138] U. Tariq, M. Hong, and W. Kim, “Quick Fix, an expeditious approach to diminish SPAM,” in *9th International Multitopic Conference, IEEE INMIC 2005*, pp. 1–4, 2005.

[139] Z. Zhong, L. Ramaswamy, K. Li, Zhenyu Zhong, and Kang Li, “ALPACAS: A Large-Scale Privacy-Aware Collaborative Anti-Spam System,” in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 556–564, 2008.

[140] A. Khorsi, “An Overview of Content-Based Spam Filtering Techniques.,” *Informatica*, vol. 31, no. 3, pp. 269–277, 2007.

[141] Y. Song, A. Kolcz, and C. L. Giles, “Better Naive Bayes classification for high-precision spam detection,” *SOFTWARE-PRACTICE & EXPERIENCE*, vol. 39, no. 11, pp. 1003–1024, Aug. 2009.

[142] C. Chen, Y. Tian, and C. Zhang, “Spam filtering with several novel bayesian classifiers,” in *19th International Conference on Pattern Recognition, 2008. ICPR 2008.*, pp. 1897–1900, 2008.

[143] Z. Yang, X. Nie, W. Xu, and J. Guo, “An approach to spam detection by naive Bayes ensemble based on decision induction,” in *Conference on Intelligent Systems Design and Applications, 2006. ISDA ’06. Sixth International*, pp. 861– 866, 2006.

[144] J. S. Kong, B. A. Rezaei, N. Sarshar, V. P. Roychowdhury, and P. O. Boykin, “Collaborative spam filtering using e-mail networks,” *Computer*, vol. 39, no. 8, pp. 67 – 73, 2006.

[145] T. Lynam, G. Cormack, and D. Cheriton, “On-line spam filter fusion,” in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 123–130, 2006.

[146] C.-H. Wu, “Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks,” *Expert Syst. Appl.*, vol. 36, no. 3, pp. 4321–4330, 2009.

[147] G. Caruana and M. Li, “A survey of emerging approaches to spam filtering,” *ACM Computing Surveys*, vol. 44, no. 2, pp. 1–27, Feb. 2012.

[148] “PoweredBy - Hadoop Wiki.” [Online]. Available: http://wiki.apache.org/hadoop/PoweredBy.

[149] J. Dean and S. Ghemawat, “MapReduce,” *Communications of the ACM*, vol. 51, no. 1, p. 107, Jan. 2008.

[150] “RSA Conference | Connect | HT1-301: Yokai vs. the Elephant: Hadoop and the Fight Against Shape-Shifting Spam (PK Session).” [Online]. Available: http://365.rsaconference.com/community/connect/blog/2010/02/12/ht1-301-yokai-vs-the-elephant-hadoop-and-the-fight-against-shape-shifting-spam-pk-session. [Accessed: 28-Jul-2012].

[151] K. Woodsend and J. Gondzio, “Hybrid MPI/OpenMP Parallel Linear Support Vector Machine Training,” *Journal Of Machine Learning Research*, vol. 10, pp. 1937–1953, Aug. 2009.

[152] T.-N. Do and F. Poulet, “Classifying one billion data with a new distributed svm algorithm,” in *2006 International Conference onResearch, Innovation and Vision for the Future*, pp. 59–66, 2006.

[153] C.-Y. Chiu and Y.-T. Huang, “Integration of Support Vector Machine with Na&#239;ve Bayesian Classifier for Spam Classification,” in *Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery - Volume 01*, pp. 618–622, 2007.

[154] S. Kang, S. Lee, J. Kim, and I. Nam, “Two phase approach for spam-mail filtering,” *CIS’04: Proceedings of the First international conference on Computational and Information Science*, pp. 800–805, 2005.

[155] B. C. Kuszmaul, “Cilk provides the ‘best overall productivity’ for high performance computing,” in *Proceedings of the nineteenth annual ACM symposium on Parallel algorithms and architectures - SPAA ’07*, p. 299, 2007.

[156] J. Yang, X. Yang, and J. Zhang, “A Parallel Multi-Class Classification Support Vector Machine Based on Sequential Minimal Optimization,” in *First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS’06)*, pp. 443–446, 2006.

[157] T.-N. Do, V.-H. Nguyen, and F. Poulet, *Advanced Data Mining and Applications*, vol. 5139. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 147–157, 2008.

[158] C.-T. T. Chu, S. K. Kim, Y.-A. A. Lin, Y. Yu, G. R. Bradski, A. Y. Ng, and K. Olukotun, “Map-Reduce for Machine Learning on Multicore,” in *Neural Information Processing Systems*, pp. 281–288, 2006.

[159] H. P. Graf, E. Cosatto, L. L. Bottou, I. Durdanovic, and V. Vapnik, “Parallel support vector machines: The cascade svm,” *IN ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*, pp. 521–528, 2005.

[160] C. Xu and Y. Zhou, “Transductive Support Vector Machine for Personal Inboxes Spam Categorization,” in *CISW ’07 Proceedings of the 2007 International Conference on Computational Intelligence and Security Workshops*, pp. 459 –463, 2007.

[161] Osterman, “Emerging Trends in Fighting Spam - An Osterman Research White Paper.” Symantec, Jun-2007.

[162] Y. Gao, M. Yang, X. Zhao, and B. Pardo, “Image spam hunter,” in *ICASSP ’08. IEEE International Conference onAcoustics, Speech and Signal Processing*, vol. 5678/2009, pp. 1765 – 8, 2008.

[163] B. Biggio, G. Fumera, I. Pillai, and F. Roli, “Image spam filtering by content obscuring detection, Fourth conference on email and antispam,” in *CEAS ’07 Fourth Conference on Email and Anti-Spam*, 2007.

[164] H. Zuo, W. Hu, O. Wu, Y. Chen, and G. Luo, “Detecting image spam using local invariant features and pyramid match kernel,” *WWW ’09 Proceedings of the 18th international conference on World wide web*, pp. 1187–1188, 2009.

[165] B. Mehta, S. Nangia, M. Gupta, and W. Nejdl, “Detecting image spam using visual features and near duplicate detection,” *WWW ’08 Proceedings of the 17th international conference on World Wide Web*, pp. 497–506, 2008.

[166] T. Joachims, “Text Categorization with Suport Vector Machines: Learning with Many Relevant Features,” pp. 137–142, Apr. 1998.

[167] G. Huang, K. Z. Mao, C. Siew, D. Huang, and S. Member, “Fast modular network implementation for support vector machines,” *IEEE Transactions on Neural Networks*, vol. 16, pp. 1651–1663, 2005.

[168] E. Y. Chang, K. Zhu, H. Wang, H. Bai, J. Li, and Z. Qiu, “PSVM: Parallelizing Support Vector Machines on Distributed Computers,” in *Processing Systems*, vol. 20, 2007.

[169] J. A. K. Suykens and J. Vandewalle, “Least Squares Support Vector Machine Classifiers,” *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, Jun. 1999.

[170] H. Yu, J. Yang, and J. Han, “Classifying large data sets using SVMs with hierarchical clusters,” in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD ’03*, p. 306, 2003.

[171] T. Joachims, “Svm-light support vector machine, 2002,” *URL= http://svmlight. joachims. org*, 2009.

[172] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, Apr. 2011.

[173] L. Bottou, A. Bordes, and S. Ertekin, “Fast Kernel Classifiers with Online and Active Learning,” *Journal of Machine Learning Research*, vol. 6, no. 9, pp. 1579–1619, 2005.

[174] E. Osuna, R. Freund, and F. Girosi, “An improved training algorithm for support vector machines,” in *Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*, pp. 276–285, 1997.

[175] G. B. Arfken and H. j. Weber, *Lagrange Multipliers*, 5th ed. Elsevier Academic, 2005.

[176] J. Kim and S. Kang, “Feature selection by fuzzy inference and its application to spam-mail filtering,” *Computational Intelligence and Security*, pp. 361–366, 2005.

[177] H. Wang, Y. Yu, and Z. Liu, “SVM classifier incorporating feature selection using GA for spam detection,” *Embedded and Ubiquitous Computing–EUC 2005*, pp. 1147–1154, 2005.

[178] K. Taura, K. Kaneda, T. Endo, and A. Yonezawa, “Phoenix,” *ACM SIGPLAN Notices*, vol. 38, no. 10, p. 216, Oct. 2003.

[179] “Welcome to Apache Hadoop!” [Online]. Available: http://hadoop.apache.org/. [Accessed: 09-May-2011].

[180] O. O’Malley, “Apache hadoop wins terabyte sort benchmark.” [Online]. Available: http://developer.yahoo.net/blogs/hadoop/2008/07/apache_hadoop_wins_terabyte_sort_benchmark.html. [Accessed: 11-Feb-2013].

[181] O. O’Malley, “Terabyte Sort on Apache Hadoop,” *Yahoo, available online at: http://sortbenchmark. org/Yahoo-Hadoop. pdf*, 2008. [Online]. Available: sortbenchmark.org/YahooHadoop.pdf. [Accessed: 23-Aug-2010].

[182] V. Vapnik and A. Lerner, “Pattern Recognition using Generalized Portrait Method,” *Automation and Remote Control*, vol. 24, 1963.

[183] M. Kearns, “Efficient noise-tolerant learning from statistical queries,” *Journal of the ACM*, vol. 45, no. 6, pp. 983–1006, Nov. 1998.

[184] G. Caruana, M. Li, and M. Qi, “A MapReduce based parallel SVM for large scale spam filtering,” in *Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on*, vol. 4, pp. 2659–2662, 2011.

[185] Y. Liu, M. Li, S. Hammoud, N. K. Alham, and M. Ponraj, “A MapReduce based distributed LSI,” in *Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on*, vol. 6, pp. 2978–2982, 2010.

[186] N. K. Alham, M. Li, Y. Liu, and S. Hammoud, “A MapReduce-based distributed SVM algorithm for automatic image annotation,” *Computers & Mathematics with Applications*, vol. 62, no. 7, pp. 2801–2811, Oct. 2011.

[187] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the fifth annual workshop on Computational learning theory - COLT ’92*, pp. 144–152, 1992.

[188] N. A. Syed, S. Huan, L. Kah, and K. Sung, “Incremental Learning with Support Vector Machines,” 1999. [Online]. Available:

http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.46.6367. [Accessed: 03-Aug-2010].

[189] Y.-M. Wen and L. Bao-Liang, “Incremental Learning of SVM’s by Classifier Combining,” in *PAKDD’07 Proceedings of the 11th Pacific-Asia conference on Advances in knowledge discovery and data mining*, 2007.

[190] D. Caragea, A. Silvescu, and V. Honavar, “Incremental and distributed learning with support vector machines,” in *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, p. 1067, 2000.

[191] A. Asuncion and D. J. Newman, *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2007.

[192] A. Pavlo, E. Paulson, and A. Rasin, “A comparison of approaches to large-scale data analysis,” in *SIGMOD ’09 Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pp. 165–178, 2009.

[193] “5 Common Questions About Hadoop « Cloudera » Apache Hadoop for the Enterprise.” [Online]. Available: http://www.cloudera.com/blog/2009/05/5-common-questions-about-hadoop/. [Accessed: 30-Apr-2010].

[194] Y. Liu, M. Li, N. K. Alham, and S. Hammoud, “HSim: A MapReduce simulator in enabling Cloud Computing,” *Future Generation Computer Systems*, vol. 29, no. 1, pp. 300–308, Jan. 2013.

[195] A. Frank and A. Asuncion, *UCI Machine Learning Repository - Adult Data Set*. University of California, Irvine, School of Information and Computer Sciences, 2010.

[196] H.-N. H.-J. Kim, J. J. Jung, and G. Jo, “On Enhancing the Performance of Spam Mail Filtering System Using Semantic Enrichment,” in *AI’04 Proceedings of the 17th Australian joint conference on Advances in Artificial Intelligence*, pp. 1095–1100, 2004.

[197] W. Li, N. Zhong, Y. Y. Yao, J. Liu, and C. Liu, “Spam filtering and email-mediated applications,” in *WImBI’06 Proceedings of the 1st WICI international conference on Web intelligence meets brain informatics*, pp. 382–405, 2007.

[198] M. Sewell, “Ensemble Methods,” 2010. [Online]. Available: http://machine-learning.martinsewell.com/ensembles/. [Accessed: 03-Jul-2010].

[199] H. Yang and J. Callan, “Ontology generation for large email collections,” in *Proceedings of the 2008 international conference on Digital government research*, pp. 254–261, 2008.

[200] K. Taghva, J. Borsack, J. Coombs, A. Condit, S. Lumos, and T. Nartker, “Ontology-based Classification of Email,” in *ITCC ’03 Proceedings of the International Conference on Information Technology: Computers and Communications*, p. 194, 2003.

[201] A. Tsymbal, “The Problem of Concept Drift: Definitions and Related Work,” Citeseer, Ireland, Apr. 2004.

[202] A. Han, H.-J. Kim, I. Ha, and G.-S. Jo, “Semantic Analysis of User Behaviors for Detecting Spam Mail,” in *2008 IEEE International Workshop on Semantic Computing and Applications*, pp. 91–95, 2008.

[203] K. Gee, “Using latent semantic indexing to filter spam,” in *SAC ’03 Proceedings of the 2003 ACM symposium on Applied computing*, pp. 460–464, 2003.

[204] D. Brewer, S. Thirumalai, K. Gomadam, and K. Li, “Towards an ontology driven spam filter,” in *ICDEW ’06 Proceedings of the 22nd International Conference on Data Engineering Workshops*, pp. 79 –79, 2006.

[205] G. Gargiulo, A. Picariello, and C. Sansone, “Using Visual and Semantic features for anti-spam filtering,” *mls-nips07.first.fraunhofer.de*, 2007. [Online]. Available: http://mls-nips07.first.fraunhofer.de/abstracts/12-Gargiulo.pdf. [Accessed: 03-Jul-2010].

[206] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, Mar. 2003.

[207] G. Wilfried, J. Andreas, and R. Neumayer, “Spam filtering based on latent semantic indexing,” in *Survey of Text Mining II*, Springer, pp. 165–183, 2008.

[208] L. Breiman, “Bagging predictors,” *Machine Learning*, vol. 24, no. 2, pp. 123–140, Aug. 1996.

[209] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, Jan. 2002.

[210] I. Horrocks, P. F. Patel-Schneider, and F. Van Harmelen, “From SHIQ and RDF to OWL: The Making of a Web Ontology Language,” *Journal of Web Semantics*, vol. 1, p. 2003, 2003.

[211] C. Lee, S. Park, D. Lee, J. Lee, O.-R. Jeong, and S. Lee, “A comparison of ontology reasoning systems using query sequences,” in *ICUIMC ’08 Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pp. 543–546, 2008.

[212] W3C, “SWValidators - Semantic Web Standards.” [Online]. Available: http://www.w3.org/2001/sw/wiki/SWValidators. [Accessed: 03-Jul-2010].

[213] D. McGuinness and F. Van Harmelen, “OWL web ontology language overview,” *W3C recommendation*, 2004.

[214] G. Caruana, M. Li, and Y. Liu, “An Ontology Enhanced Parallel SVM for Scalable Spam Filter Training,” *Neurocomputing*, no. 108, pp. 45–57, 2013.

[215] E. Sirin, B. Parsia, B. Grau, A. Kalyanpur, and Y. Katz, “Pellet: A practical OWL-DL reasoner,” *Software Engineering and the Semantic Web*, vol. 5, no. 2, pp. 51–53, Jun. 2007.

[216] E. Prud'hommeaux and A. Seaborne, "SPARQL Query Language for RDF (Working Draft)," Mar. 2007.

[217] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson, "Jena: implementing the semantic web recommendations," in *WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pp. 74–83, 2002.

[218] S. C. for B. I. Research, "Protege." [Online]. Available: http://protege.stanford.edu/. [Accessed: 01-Oct-2011].

[219] U. Rebbapragada and C. E. Brodley, *Machine Learning: ECML 2007*, vol. 4701. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 708–715, 2007.

[220] E. M. Voorhees and L. P. Buckland, Eds., "Proceedings of The Sixteenth Text REtrieval Conference, TREC 2007, Gaithersburg, Maryland, USA, November 5-9, 2007," vol. Special Pu, 2007.

[221] T. D. Braun, H. J. Siegel, N. Beck, L. L. Bölöni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, B. Yao, D. Hensgen, and R. F. Freund, "A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems," *Journal of Parallel and Distributed Computing*, vol. 61, no. 6, pp. 810–837, Jun. 2001.

[222] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems," *Journal of Scheduling*, vol. 12, no. 4, pp. 417–431, Oct. 2008.

[223] J. J. Dongarra, E. Jeannot, E. Saule, and Z. Shi, "Bi-objective scheduling algorithms for optimizing makespan and reliability on heterogeneous systems," in *Proceedings of the nineteenth annual ACM symposium on Parallel algorithms and architectures - SPAA '07*, p. 280, 2007.

[224] J. Yang, H. Xu, L. Pan, P. Jia, F. Long, and M. Jie, “Task scheduling using Bayesian optimization algorithm for heterogeneous computing environments,” *Applied Soft Computing*, vol. 11, no. 4, pp. 3297–3310, Jun. 2011.

[225] Z. Shi and J. J. Dongarra, “Scheduling workflow applications on processors with different capabilities,” *Future Generation Computer Systems*, vol. 22, no. 6, pp. 665–675, May 2006.

[226] S. Babu, “Towards automatic optimization of MapReduce programs,” *Proceedings of the 1st ACM symposium on Cloud computing - SoCC ’10*, p. 137, Jun. 2010.

[227] Y. Kwon, M. Balazinska, B. Howe, and J. Rolia, “SkewTune,” in *Proceedings of the 2012 international conference on Management of Data - SIGMOD ’12*, p. 25, 2012.

[228] E. Bortnikov, A. Frank, E. Hillel, and S. Rao, “Predicting execution bottlenecks in map-reduce clusters,” in *HotCloud’12: Proceedings of the 4th USENIX conference on Hot Topics in Cloud Ccomputing*, p. 18, 2012.

[229] J. Xiao and Z. Xiao, “High-integrity mapreduce computation in cloud with speculative execution.,” in *Theoretical and mathematical foundations of computer science. Second international conference, ICTMF 2011, Singapore, May 5--6, 2011. Selected papers.*, Berlin: Springer, pp. 397–404, 2011.

[230] D. J. Lilja and B. Hamidzadeh, “Dynamic task scheduling using online optimization,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 11, pp. 1151–1163, 2000.

[231] T. Sandholm and K. Lai, “Dynamic proportional share scheduling in Hadoop,” in *JSSPP’10 Proceedings of the 15th international conference on Job scheduling strategies for parallel processing*, pp. 110–131, 2010.

[232] K. Kc and K. Anyanwu, “Scheduling Hadoop Jobs to Meet Deadlines,” in *2010 IEEE Second International Conference on Cloud Computing Technology and Science*, pp. 388–392, 2010.

[233] M. M. Rafique, B. Rose, A. R. Butt, and D. S. Nikolopoulos, “Supporting MapReduce on large-scale asymmetric multi-core clusters.,” *Operating Systems Review*, vol. 43, no. 2, pp. 25–34, 2009.

[234] C. Tian, H. Zhou, Y. He, and L. Zha, “A Dynamic MapReduce Scheduler for Heterogeneous Workloads,” in *2009 Eighth International Conference on Grid and Cooperative Computing*, pp. 218–224, 2009.

[235] D. Gillick, A. Faria, and J. DeNero, “MapReduce: Distributed computing for machine learning,” 2006. [Online]. Available: http://cs.smith.edu/dftwiki/images/6/68/MapReduceDistributedComputingMachineLearning.pdf. [Accessed: 11-Feb-2013].

[236] Z. Ma, L. Gu, and L. G. Zhiqiang Ma, “The limitation of MapReduce: A probing case and a lightweight solution,” in *In Proc. of the 1st Intl. Conf. on Cloud Computing, GRIDs, and Virtualization*, pp. 68–73, 2010.

[237] Z. Guo and G. Fox, “Improving MapReduce Performance in Heterogeneous Network Environments and Resource Utilization,” in *CCGRID*, pp. 714–716, 2012.

[238] G. Lee, B.-G. Chun, and H. Katz, “Heterogeneity-aware resource allocation and scheduling in the cloud,” in *HotCloud’11 Proceedings of the 3rd USENIX conference on Hot topics in cloud computing*, p. 4, 2011.

[239] M. Zaharia, A. Konwinski, A. D. Joseph, R. H. Katz, and I. Stoica, “Improving MapReduce Performance in Heterogeneous Environments.,” in *OSDI*, pp. 29–42, 2008.

[240] B. T. Rao, N. V. Sridevi, V. K. Reddy, and L. S. S. Reddy, “Performance Issues of Heterogeneous Hadoop Clusters in Cloud Computing,” *Global Journal of Computer Science and Technology*, vol. XI, no. VIII, p. 6, Jul. 2012.

[241] “Network performance within Amazon EC2 and to Amazon S3 | RightScale Blog on WordPress.com.” [Online]. Available:

http://blog.rightscale.com/2007/10/28/network-performance-within-amazon-ec2-and-to-amazon-s3/. [Accessed: 30-Jan-2013].

[242] M. B. YongChul Kwon, Y. Kwon, M. Balazinska, B. Howe, and J. Rolia, “A Study of Skew in MapReduce Applications,” in *Proceedings of the 2012 international conference on Management of Data - SIGMOD ’12*, p. 25, 2012.

[243] S. Shivle, P. Sugavanam, H. J. Siegel, A. A. Maciejewski, T. Banka, K. Chindam, S. Dussinger, A. Kutruff, P. Penumarthy, P. Pichumani, P. Satyasekaran, D. Sendek, J. Smith, J. Sousa, J. Sridharan, and J. Velazco, “Mapping subtasks with multiple versions on an ad hoc grid,” *Parallel Computing*, vol. 31, no. 7, pp. 671–690, Jul. 2005.

[244] S. Ali, H. J. Siegel, M. Maheswaran, S. Ali, and D. Hensgen, “Task Execution Time Modeling for Heterogeneous Computing Systems,” in *HCW ’00 Proceedings of the 9th Heterogeneous Computing Workshop*, p. 185, 2000.

[245] M. I. Daoud and N. Kharma, “A hybrid heuristic–genetic algorithm for task scheduling in heterogeneous processor networks,” *Journal of Parallel and Distributed Computing*, vol. 71, no. 11, pp. 1518–1531, Nov. 2011.

[246] Z. Ou, H. Zhuang, J. K. Nurminen, A. Ylä-Jääski, and P. Hui, “Exploiting hardware heterogeneity within the same instance type of Amazon EC2,” p. 4, Jun. 2012.

[247] “Amazon EC2 Instance Types.” [Online]. Available: http://aws.amazon.com/ec2/instance-types/. [Accessed: 03-Jan-2013].

[248] B. Palanisamy, A. Singh, L. Liu, and B. Jain, “Purlieus,” in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis on - SC ’11*, p. 1, 2011.

[249] S. Kavulya, J. Tan, R. Gandhi, and P. Narasimhan, “An Analysis of Traces from a Production MapReduce Cluster,” in *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 94–103, 2010.

[250] B. T. Rao and L. S. S. Reddy, “Scheduling Data Intensive Workloads through Virtualization on MapReduce based Clouds,” *International Journal of Distributed and Parallel Systems*, vol. 3, no. 4, pp. 99–110, 2012.

[251] B. T. Rao and L. S. S. Reddy, “Survey on Improved Scheduling in Hadoop MapReduce in Cloud Environments,” *International Journal of Computer Applications*, vol. 34, no. 9, p. 5, Jul. 2011.