SEMANTIC WEB MINING OF UNSTRUCTURED DATA

A Thesis

Submitted in fulfillment of the
Requirements for the award of the degree of

Doctor of Philosophy

Submitted By

Manoj Manuja (Reg. No. 950903004)

Under the supervision of

Dr. Deepak Garg

Associate Professor & Head
Computer Science and Engineering Department
Thapar University
Patiala - 147004

COMPUTER SCIENCE & ENGINEERING DEPARTMENT

Thapar University, Patiala

October, 2014

This Dissertation is dedicated

To

All my respected Gurus

Who guided me through this knowledge journey.

CERTIFICATE

I hereby certify that the work which is being presented in this thesis entitled **SEMANTIC WEB MINING OF UNSTRUCTURED DATA**, in fulfillment of the requirements for the award of degree of **DOCTOR OF PHILOSOPHY** submitted in Computer Science and Engineering Department (CSED), Thapar University, Patiala, Punjab, India, is an authentic record of my work carried out under the supervision of **Dr. Deepak Garg**, and refers the work of other researchers, which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(Manoj Manuja)

(Registration No. 950903004)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr. Deepak Garg)

Associate Professor and Head,

Computer Science & Engineering Department.

Thapar University, Patiala

July, 2014

iii

Acknowledgement

Guru govind dou khade | kake lagu pau |

Bali hari guru aapne | govind diyo batay |

"Teacher is greater than GOD (in spiritual form).. because teacher is the person who guides us and explains the pathway to reach to that eternal power"

I am indebted to all my GURUs while I write this acknowledgement page. ***Dr. Deepak Garg****, my supervisor during this research work, has been a constant source of inspiration for me. He is one person whom I can look for guidance, help and support. He is a person without attitude, full of energy and compassion. This makes him separate from the rest! I wish him all the very best in his upcoming life and profession!!!*

*Next, I would like to formally put "****RRAM****" on board… my family i.e. my wife* ***R****ashmi, my sons* ***R****ishabh and* ***A****tal with* ***m****e. Their unconditional support for me to pursue my doctoral ambitions has been phenomenal. I am really lucky to have them around me!*

Last but not the least, I thank the Almighty for reasons too numerous to mention.

Manoj Manuja

Abstract

Over the last couple of decades, web classification has gradually transitioned from syntax to semantic centered approach that classifies the text based on domain ontologies. These ontologies are either built manually or populated automatically using machine learning techniques. Pre-requisite condition to build such system is the availability of ontology which may be either full-fledged domain ontology or a seed ontology that can be enriched automatically. This is a dependency condition for any given semantic based text classification system.

We have designed, developed and implemented a web classification system that is self-governed in terms of ontology population and does not require any pre-built ontology either full-fledged or seed. It starts from user query, build a seed ontology from it and automatically enrich it by extracting concepts from the downloaded documents only.

The evaluated parameters like precision (85%), accuracy (86%), AUC (Convex) and MCC (High + ive) provide a better worth of the proposed system when compared with similar automated text classification systems.

We have used Support Vector Machines (SVMs) to find similarity / dissimilarity measures among concepts and features so that similar concepts are linked together for optimal knowledge discovery. The learning system we have developed above has two components – kernel machine for encapsulating the learning task and kernel function for imbibing the learning hypothesis. Linear kernel function has been used which primary exploits syntactic structures of the text. To improve the scope of knowledge extraction, we have exploited semantic kernel functions which use a-priori semantic information for knowledge extraction.

Therefore, building the classification system with semantic kernel functions instead of linear kernel functions forms the next step of our research. We have tried to validate

the performance and accuracy parameters obtained above by way of using semantic kernel function in place of linear kernel function. This also provides us an opportunity to explore the usefulness of semantic kernel functions in the context of semantic web mining. The evaluated parameters like precision (89.2%), accuracy (88%), AUC (More Convex Area) and MCC (More High +ve) clearly validate our framework with improved performance and accuracy measurements when we use semantic kernel functions instead of linear kernel functions.

There are a few open issues like fine tuning of query manager in our framework, use of OWL instead of RDF, and performance improvement of overall system which need to be explored more in depth as future directions to this research work.

Contents

List of Figures

List of Tables

Chapter – 1

Introduction

Last few years have seen growing recognition of information as a key business tool for the organizations to be successful. The organizations which effectively gather, analyze, understand, and act upon the information are definite winners in this new “information age”. Further to this, the realization of “web” has critically changed the perspective of how the organizations extract information from the available data in today’s world of dynamic business.

Therefore, the most important differentiator between a successful and a failed business is how an organization manages its data. The critical aspect in today’s business scenario is how data is converted into Information and subsequently how information is converted into knowledge [88, 94].

1.1 Data, Information and Knowledge

Data

Data may be defined as a collection of facts from which conclusions may be drawn. Data can be processed to create useful information.

Information

The manipulated and processed form of data is called information. It is more meaningful than data and is used for decision making. Data is used as an input for processing and

output of this processing is information. Information consists of facts and data organized to describe a particular situation or condition. Information can be either true or untrue, and is abstract.

Knowledge

Knowledge consists of facts, truths and beliefs, perspectives and concepts, judgments and expectations, methodologies and know-how of things. Knowledge is accumulated and integrated and held over a period of time to handle specific situations and challenges.

We use knowledge to determine what a specific situation means. Knowledge is applied to interpret information about the situation and to decide how to handle it. Knowledge is a belief that is true, justified, and relies on no false theories.

1.2 Data and its types

Data may be categorized in terms of its processing:

Structured data: Data that resides in fixed fields within a record or file is generally referred to as structured data. In other words, structured data is the data that has a data model. Relational databases and spreadsheets are examples of structured data.

Un-Structured Data: It refers to (usually) computerized information that either does not have a data model or has one that is not easily usable by a computer program. In other words, data with some form of structure may be characterized as unstructured if its structure is not helpful for the

desired processing task. Examples of "unstructured data" may include audio, video, and text such as the body of an e-mail message, web page, or word document.

Large organizations generate huge amount of data every day. This data is generated in different forms by different people at different times through different sources spread across different geographies. We have different types of Data generated by different stake holders in an organization [3], for example: Business Data, Marketing / Sales Data, Production Data, Project specific Data, Research and Development Data, Knowledge Management Data, Operational Data, Human Resource Data and Historical Data to name a few.

1.3 Business dilemma in a large organization

It is very crucial to extract knowledge from un-structured data which is available in various formats and generated by heterogeneous sources across a big organization. According to projections from Gartner, white- collar workers will spend anywhere from 30 to 40 percent of their time this year managing documents, up from 20 percent of their time a decade earlier. Similarly, Merrill Lynch estimates that more than 85 percent of all business information exists as unstructured data.

Sometimes, it becomes very difficult to extract knowledge from the un-structured data even after using proven algorithms and patterns [54, 86]. With the exponential growth of the internet, most of the data is now available over the web in the form of unstructured data.

I have tried to address this business requirement of knowledge extraction from un-structured data using the concept

of semantic web mining and suggested a Knowledge Driven Information Management System (KDIMS).

1.4 Motivation

Our work on this research topic has been primarily motivated by four prime factors:

Big Content: The unstructured side of the Big Data

The enterprises are slowly coming out of the age of information overload and getting comfortable with managing massive amounts of data, content and information. On one side, the pace of information creation continues to accelerate, but on the other side, a lot of IT enterprise infrastructure management systems have been evolved to manage this explosion of information generation across the enterprise. Now, big data is considered as a blessing rather than a curse. Gartner analysts predict unstructured data will grow a whopping 800 percent over the next five years, and that unstructured side of the big data content constitutes around eighty percent of an organization's total information assets. Within the organization, this unstructured data takes many forms like business documents, emails and web content. Finding insight and intelligence from this big chunk of unstructured data is first motivation for our research work.

Semantic based instead of syntax based classification

We can easily figure out through Gartner's Hype Cycle for Big Data, 2012 [35, 36], as shown in Figure – 1.1 that semantic web (SW) is going to play a major role in extracting

information, knowledge and business intelligence from unstructured side of big data content.

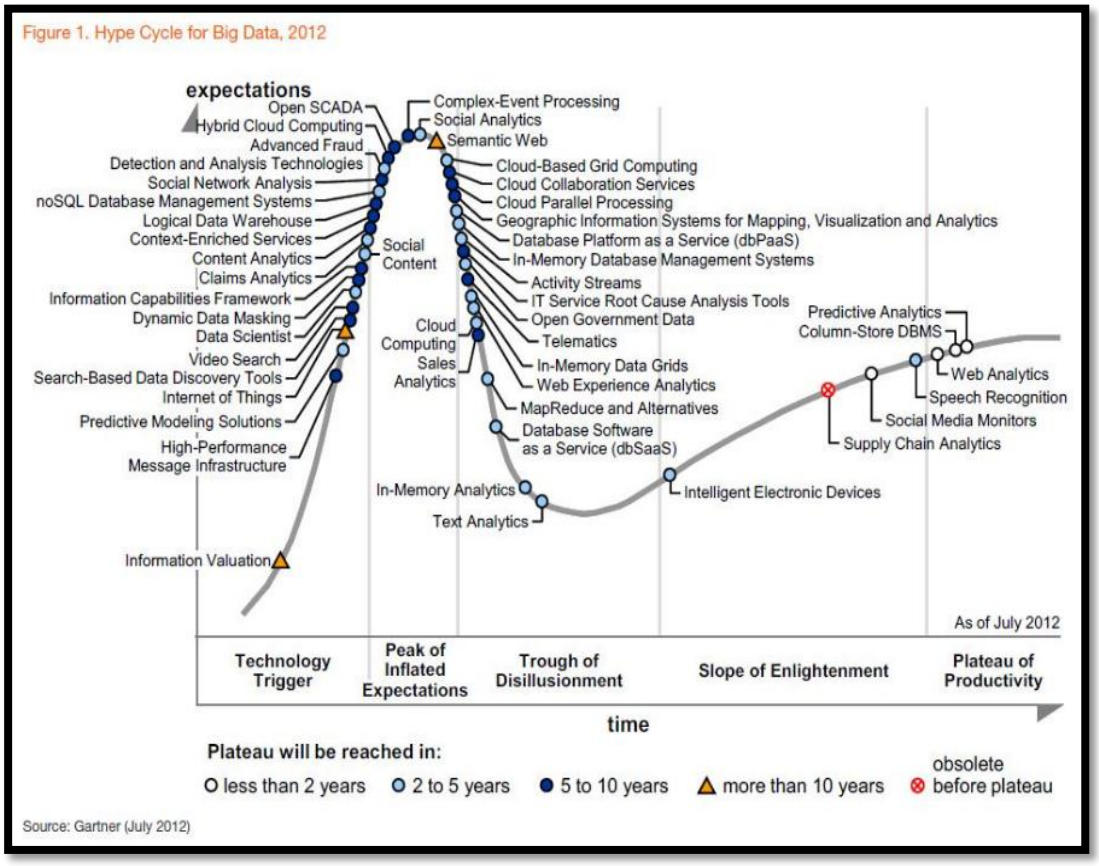


Figure – 1.1: Hype Cycle for Big Data, 2012 by Gartner

World Wide Web contains the biggest and the most current source of information on any and every domain on this earth. This set of information may contain documents ranging from best practices, technical reports, customer feedback and product review comments to name a few. Around 80% of this information is written in natural language and unstructured in nature [17]. Many a times, a novice user finds it very difficult to search for useful information on web without prior knowledge of the subject supported by rich clues. This is primarily because of the fact that web classification

systems (mainly search engines) respond to any user query on the basis of its syntax instead of searching on the basis of its semantic.

Computing for human experience (CHE) is one of the futuristic thoughts which provide a vision for future man-machine interfaces with a realistic implementation [79]. The CHE talks about technology rich intelligent systems enabling human experience to gather and apply knowledge in relevant fields of sentiment and opinion mining. CHE vision motivates authors to design and develop an intelligent system which can help to classify data available on the web with minimal explicit effort being put by the humans. In today's world of semantic web, it is indeed relevant to have any intelligent classification system based on semantics rather than syntaxes [33].

SW is a place where knowledge intensive manipulations on complex web data are performed by computing machines. For SW to function, computers must have access to collections of structured information and the sets of inference rules that machines can use to conduct automatic reasoning on a given set of data [37, 38, 39]. This motivates us to build a system which a machine can understand without any human intervention.

Existing text classification systems

A lot of work has been done on developing semantic based classification systems during the past few years [41, 63, 68, 71, 73, 80]. These systems make all forms of information linked through semantics so that human can utilize this wealth of information automatically using machine learning methods

and algorithms. A rapid growth of linked data in recent past has led to availability of domain specific ontologies on web in abundance [9, 42]. Ontologies provide a controlled vocabulary of concepts along with their relations explicitly defined using machine process-able semantics [83]. It is very time consuming for humans to build a machine understandable relationship graph covering all the domains available around us. Hence, it is imperative to automate ontology learning process which in turn helps in enhancing human-machine interaction [14, 95]. Quite a few frameworks are available which address this automation problem and most of them use full-fledged pre-built domain or seed ontology to start with the automation and enrichment process. No classification system is available (to the best of our knowledge) that is self-governed in terms of ontology population and does not require any pre-built ontology either full-fledged or seed.

Semantic Kernel based learning

The real power of the SW is to create many programs that collect content from diverse web sources, process the gathered information according to defined set of rules and share the results with other programs. In SW, these web agents share common understanding by exchanging ontologies which provide a vocabulary needed for taking meaningful decisions.

Inductive classifiers with kernel methods are extensively used for solving such learning problems [28]. Inductive learning is advantageous compared to deductive learning in SW search because it can handle inconsistencies, noise and incompleteness in SW knowledge bases well [27]. Kernel methods are particularly suited in this scenario [96].

Kernel-based learning algorithms like SVMs have become protuberant frameworks for using the ‘a-priori’ knowledge about the domain in question by means of a particular choice of the employed kernel functions [74]. SVMs can be “kernelized” for various tasks pertaining to classification and regression (i.e. supervised learning tasks) [75]. This motivates us to incorporate a-priori knowledge while building a classification system for unstructured text.

These are four motivational points that have encouraged us to take up the research work of designing and developing a knowledge Driven Information Management System (KDIMS) which should be self-governed with almost no human interface.

1.5 Problem Statement

The management of unstructured data is recognized as one of the major unsolved problems in the information technology (IT) industry. The main reason behind this problem are the tools and techniques that have proved so successful transforming structured data into business intelligence and actionable information, simply don't work when it comes to unstructured data. New approaches are necessary.

Most of the business information exists in the form of unstructured data – commonly appearing in e-mails, memos, notes from call centers and support operations, news, user groups, chats, reports, letters, surveys, white papers, marketing material, research, presentations and Web pages to name a few.

Sometimes it is very important and critical to generate meaningful information from this data for the success of the organization. With the exponential growth of the internet, most of the data is now available over web in the form of unstructured documents which is readily available for intelligent exploration and knowledge extraction.

My research work is focused towards

- Suggesting a framework which may be used to extract information from unstructured data using semantic web mining standards (OWL / RDF)
- Proposing a data model / Knowledge Driven Information management system (KDIMS) to manage the process of Information Extraction in the context of unstructured data.

Objectives:

The objectives of this research are as follows:

- Designing a framework that will extract useful information from unstructured data using Semantic web mining standards.
- Designing and developing a KDIMS which will interact with the documents to extract useful information using Semantic web mining standards.
- Validating the developed model / system
- Suggesting business usefulness of this KDIMS to the organization

1.6 Contribution towards the Research Problem

We have adopted a structured approach to explore the said research problem and have contributed as per below mentioned chapters:

Chapter – 1: Introduction

First chapter provides the introduction to the PhD topic. It gives a holistic overview of the types of data i.e. structured and unstructured with the management of unstructured data as one of the most critical ongoing problems in data management and business intelligence fields in current times. Four motivation points to work on this research problem are also discussed. Problem statement and objectives of this thesis are shared towards the end.

Chapter – 2: Literature Survey

A detailed literature survey has been done and is mainly focused towards semantic web mining, ontologies, kernel methods, Support Vector Machines (SVMs) and Semantic kernel functions. Usefulness of SVM for information extraction from unstructured data is also probed.

Chapter – 3: KDIMS Framework – Design & Development

An insight into design and development of Knowledge Driven Information Management System has been provided in this chapter. The details of the framework along with implementation steps have been shared in this chapter.

Chapter – 4: Performance & Accuracy Analysis of the proposed system

This chapter provides the complete performance and accuracy analysis of developed KDIMS. Experimental details of the implementation set-up for the proposed system have been shared.

Chapter – 5: System Validation with Semantic Kernel Functions

System developed in chapter 3 and implemented in chapter 4 is being validated by using semantic kernel functions in place of linear kernel functions. Four semantic kernels are implemented for KDIMS here. Complete details are provided in this chapter.

Chapter – 6: Conclusions and Future Directions

This chapter concludes our research assignment and provides a conclusive remarks on the experimental results along with one-on-one mapping with the objectives. This chapter also provides the future directions on the said research topic and provides the details of next course of action we intend to take up after completing the committed objectives in the beginning of the assignment.

References are given in the end to provide the details of all citations used during this research work.

Chapter – 2

Literature Survey

Keeping in view of the potential opportunity to extract knowledge from the huge amount of data available in an organization, this survey has been broadened over finding the relevant literature in the field of semantic web mining of unstructured data.

The literature survey primarily focuses around exploring various theories, frameworks and models suggested by researchers and industry professionals in the field of semantic web mining of unstructured data.

A structured approach is being adopted during this exercise of finding what has been done by the research community in the said field till date supported by the future directions and existing challenges.

Below fields / topics are explored to arrive at more clarity on the subject.

- Semantic Web Mining
- Ontologies
- Kernel Methods
- Support Vector Machines
- Semantic Kernel Functions
- Application of SVMs in Ontology Learning and Information Extraction

2.1 Semantic Web Mining

The semantic web is based on the vision of *Tim Berners-Lee* [8, 87], the inventor of the World-Wide-Web (WWW). According to him, “The semantic web is not a separate web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.”

Semantic web mining aims at combining the two emergent research areas of semantic web and web mining [4]. The idea is to improvise the results of web mining by exploiting the new semantic structures in the Web; and on the other hand, make use of web mining, for building up the semantic web by extracting useful patterns, structures, and semantic relations from existing web resources.

Mining structures from the un-structured data for semantics is an upcoming research area in the field of semantic web mining. The emergence of powerful search engines has greatly improved our ability to search for data on the web; however, such access is still primarily restricted to structured or semi-structured data. We can search for and access information available as HTML, but are not yet able to gain easy access to the hidden web. It is not easy to get to the correct web form, and even harder to find a suitable web service. When we do find the correct web form or web service, there is an additional step of understanding its schema and reformulating the user's query to fit that schema. While humans do this regularly, one form at a time, it is difficult to automate the process of query reformulation, and therefore

we cannot leverage the wealth of information residing behind web forms and services.

The techniques for exploiting corpora of documents do not apply directly to searching structured data. The main reason is that searching structured data requires understanding of the underlying semantics of the data sources. This structure is mostly (but not completely) specified by the schema. However, in specifying these semantics, the actual words used and the information organization depend more on the developer's whim, and little variations may account for very different semantics.

SW is a place where knowledge intensive manipulations on complex web data are performed by computing machines. For SW to function, computers must have access to collections of structured information and the sets of inference rules that machines can use to conduct automatic reasoning on a given set of data [37, 38, 39]. This information may be categorized as Knowledge Representation (KR).

There are primarily two important technologies for developing KR for the SW:

- XML (eXtensible Markup Language): allows users to add a capricious structure to their documents but it does not say anything about the meaning of the said structures.
- RDF (Resource Description Framework): expresses the meaning of the structure and encodes it in sets of triples.

Each triple is like an S-P-O (Subject-Predicate-Object) of an elementary sentence. These triples represent a document making assertions that particular things have specific

properties with certain values and can be written easily using XML tags. In semantic technology, the focus is generally to formulate flexible data model (called Triples) from the user friendly domain query.

The triples of RDF, in fact, form webs of information about related things. These information webs are generally silos and may or may not collaborate with each other on their own to provide meaning to the data. But the key to SW is to discover common meaning across such different databases available on web.

Ontologies provide a solution to this problem [64] as discussed in section 2.2 of this chapter. Popular KR formalism is the family of Description Logics (DLs) which allows modeling the relevant properties of a domain by means of [5, 10]:

- Classes i.e. unary predicates, denoting sets of individuals
- Object properties i.e. binary predicates, denoting binary relationships between individuals
- Data-type properties i.e. binary predicates denoting relations between individuals and specific data-types.

Terminological axioms in the ontology can combine and relate classes and their respective roles by the use of various defined concept and role constructors. Assertional axioms in associated Knowledge Bases (KBs) make statements about the domain instances. The semantics of certain DLs are

typically given as model theoretic semantics by relating the syntax of the logic and the models of a domain [10].

DLs have been well adopted as the core technology for ontology languages like Web Ontology Language (OWL) [5]. The practical importance of DLs is evident from the fact that they form the basis of the OWL [5]. As an adopted SW standard, OWL is increasingly being used to define ontologies. These ontologies in turn are gradually further used to describe data instances available on the web which are mainly people, places, and services.

The real power of the SW is to create many programs that collect content from diverse web sources, process the gathered information according to defined set of rules and share the results with other programs. In SW, these web agents share common understanding by exchanging ontologies which provide a vocabulary needed for taking meaningful decisions.

Inductive classifiers with kernel methods are extensively used for solving such learning problems [28]. Inductive learning is advantageous compared to deductive learning in SW search because it can handle inconsistencies, noise and incompleteness in SW knowledge bases well [27]. Kernel methods are particularly suited in this scenario [96].

Kernel-based learning algorithms like SVMs have become protuberant frameworks for using the ‘a-priori’ knowledge about the domain in question by means of a particular choice of the employed kernel functions [74]. SVMs [90] can be “kernelized” for various tasks pertaining to classification and regression (i.e. supervised learning tasks).

2.2 Ontologies

As generally defined by web researchers and SW engineers, ontology is a formal document or a file that defines the relations among various triples and terms available in a given web data. It has taxonomy and a set of pertinent inference rules. The taxonomy is used to define classes of objects and also relations among them. This taxonomy is further enhanced and powered by inference rules. One can use these ontologies for improving the accuracy of ‘web search’ wherein the search program can look for those only web pages that refer to similar and matching concepts instead of all the web pages with even irrelevant and ambiguous concepts also. Ontologies provide a shared and common understanding of a particular domain with a description of data instances in the form of vocabulary.

Extracting ontology from the web is a challenging task [1]. Ontology learning exploits a lot of existing resources, like text, thesauri, dictionaries, databases and so on. It combines techniques of several research areas, e. g., from machine learning, information retrieval, or agents, and applies them to discover the `semantics' in the data and to make them explicit [53].

The techniques produce intermediate results which must finally be integrated in one machine-understandable format, e. g., an ontology [55]. A few systems have already been developed by research community across the world to extract ontology [23].

Several standards such as the Resource Description Framework (RDF) [69, 70] and Web Ontology Language (OWL) [69]

have been developed to realize the layer cake of the Semantic Web. Resource Description Framework (RDF) is being used by people to represent metadata of web pages which can be processed by a machine [69]. It describes a data model to represent all relations between different resources.

The OWL Web Ontology Language is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics.

Information Retrieval on the Semantic Web

Web documents may contain free text along with some mark-up. There are many potential uses for annotation on the semantic web including workflow, image retrieval, database mediation etc. Information retrieval, supported by simple and complex question answering scenarios are the major thrust areas on which researchers are currently working [98].

The two fast-developing research areas Semantic Web and Web Mining both build on the success of the World Wide Web (WWW) [56]. They complement each other well because they each address one part of a new challenge posed by the great success of the current WWW. Most data on the Web is so unstructured that it can only be understood by humans, but the amount of data is so huge that it can only be processed efficiently by machines. The semantic web addresses the first part of this challenge by trying to make the data (also) machine-understandable, while web mining addresses the second part by

(semi-)automatically extracting the useful knowledge hidden in these data, and making it available as an aggregation of manageable proportions.

These techniques can be used for mining the semantic web itself. The wording semantic web mining emphasizes spectrum of possible interaction between both research areas: It can be read both as semantic (web mining) and as (semantic web) mining.

In the past few years, there have been many attempts at “breaking the syntax barrier” on the web [4]. A number of them rely on the semantic information in text corpora that is implicitly exploited by statistical methods. Some methods also analyze the structural characteristics of data; they profit from standardized syntax like XML. These relate the syntactic tokens to background knowledge represented in a model with formal semantics. When we use the term “semantic”, we thus have in mind a formal logical model to represent knowledge.

To supplement the above statement, here are some research directions that must be pursued to ensure that we continue to develop web mining technologies that will enable the power of www to be realized. These constitute web metrics and measurements, process mining, temporal evolution of the Web, web services optimization, fraud and threat analysis, and web mining and privacy.

Along with these research directions, there are some challenges involved in large scale integration on the web namely the realization of the mining framework, the robustness of mining techniques, and the exploration of holistic insight.

The implementation challenge will be to develop a database management system to manage the entire process of information extraction.

If we are to design such a system, how should it look? What will be the capabilities? The key challenges include data modelling and representational issues; need for newer index structures; standardization for Information Extraction (IE); data cleaning and fusion; relationships between uncertainty management in the context of IE and probabilistic databases; and finally the role of user knowledge and the iterative nature of user interaction.

2.3 Kernel Methods

Kernel methods are a family of efficient statistical learning algorithms, including SVMs [46, 76, 61, 89], that have been effectively and very efficiently applied to a variety of information retrieval (IR) tasks e.g. in domains that typically require structured representations [37, 38, 39, 76, 97].

The major paradigm behind kernel methods is to decouple employed learning algorithm from the representation of the data instances under investigation [74, 78]. Unkernalized algorithms operate on simple vectors of real numbers. These algorithms generate hypotheses that are typically tied to a geometric interpretation within the corresponding vector space. By virtue of the design of the algorithms of interest, the input vectors need not be accessible directly by them. Instead, it is adequate that they are able to access the evaluations of the inner or scalar product of two vectors x,

y in this space. Resulting hypotheses are then expressed using linear combinations of the input objects [10].

Kernel-based learning algorithms express the learned hypothesis by means of linear combinations of a specific type of similarity functions called kernel functions. The kernel function computes the similarity of data instances in such a way that it is equivalent to an inner product in some (possibly unknown) vector space [10].

There is an intriguing property of kernel-based learning algorithms which highlights that kernel functions need not be restricted on vector-type data as arguments but can be defined directly on data items of arbitrary type as long as some trivial restrictions are applied. This helps in working on heterogeneous and interconnected data directly that does not have a natural vector-style representation [78].

The learning algorithm (inductive bias) and the choice of the kernel function (language bias) are almost completely independent in a given kernel method [96]. Thus, attribute-valued instance spaces can be converted into ones suitable for structured spaces (for example graphs, trees) by simply replacing the kernel function with a suitable one in the algorithm. This motivates the increasing interest in the SVMs and other kernel methods [74] for those who would like to reproduce learning in high-dimensional spaces while still working in a vectorial representation.

Kernel methods can be used where we want to directly apply machine learning algorithms on SW-type of instance data. The

use of kernels avoids extensive efforts on pre-processing of the data.

They can be very efficient because by means of a kernel function they map the original feature space of the considered data set into a high-dimensional space which makes the learning task simplified. But in some cases, expanded space of much higher dimensionality may degrade the generalization process [23] therefore, choice of kernel method is also critical.

2.4 Support Vector Machines

Statistical learning theory is the theoretical approach to understand the learning machines in the context of learning and their ability to generalize [45, 46]. SVMs provide a framework for "learning from examples" using the theory of machine and statistics learning [7, 20]. They score over the conventional learning algorithms in terms of addressing noisy data, high dimensionality and non-Gaussian distribution of data [34, 48]. The general class of algorithms resulting from such process is known as "kernel methods" [76]. Originally, kernel methods were suggested by [62, 93].

An SVM is just like one of the abstract learning machines which learn from a training data set, attempt to generalize the learning and make correct predictions on a test data set [15, 46, 61, 85, 89]. For the training data we have a set of input vectors, denoted x_i, with each input vector having a number of component features. These input vectors are paired with corresponding labels, which we denote as y_i, and there are m such pairs (i = 1,2,…m). For a given situation, we generally

need to predict in affirmative or negative i.e. either the result is ‘yes’ or ‘no’.

The training data can be viewed as labeled data-points in an input space which is depicted in Figure – 2.1.

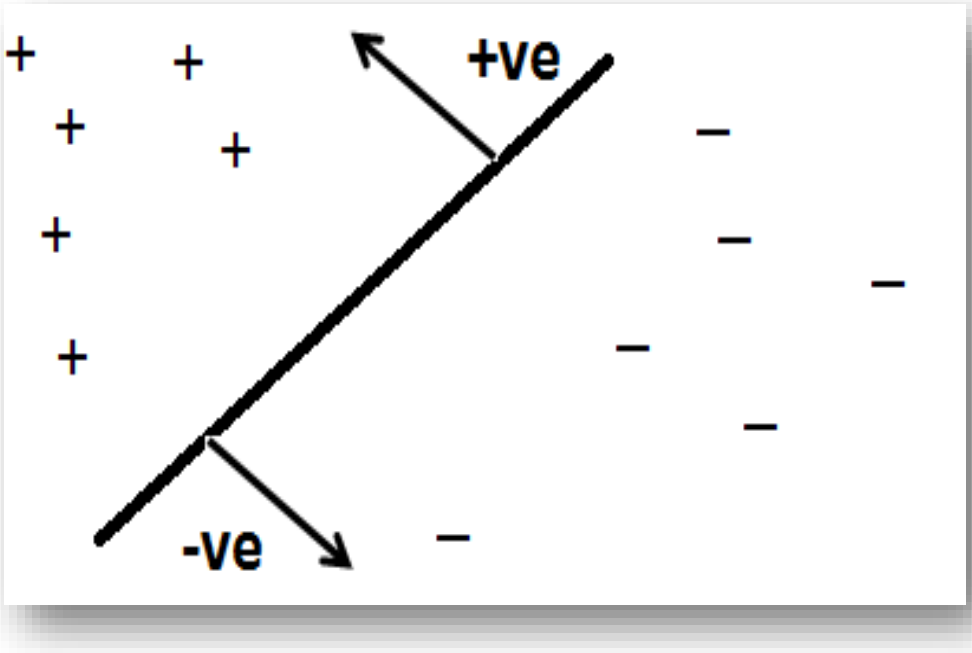


Figure – 2.1: Two Labeled Data Clusters separated by a Hyperplane

For two classes of well separated data, the learning task amounts to finding a directed hyperplane, that is, an oriented hyperplane such that data-points on one side will be labeled $y_i = +1$ and data-points on the other side as $y_i = -1$. In SVM, the directed hyperplane is intuitive: it is hyperplane that is maximally distant from the given two classes of labeled

points located on each side. The closest such points on both sides have maximum influence on the position of this separating hyperplane and are therefore call support vectors. The separating hyperplane is given as $w.x + b = 0$ (where . denotes the inner or scalar product, 'b' is the offset or the bias of the hyperplane from the origin in input space, x are points located within the hyperplane and the normal to the hyperplane, the weights w, determine its orientation).

The above case is too simple for many applications. Figure – 2.1 shows two labeled clusters which are readily separable by a hyperplane, which is simply a line in this 2-D illustration.

In real scenarios, these two clusters could be highly intermeshed with overlapping data-points: the dataset is then not linearly separable as exposed in Figure – 2.2. This situation motivates us to bring together the concept of kernels. We can also observe in the Figure – 2.2 that stray data-points could act as anomalous support vectors with a significant impact on the orientation of the hyperplane. We thus must put a mechanism in place to handle noisy and anomalous data-points which may point to multi-class data.

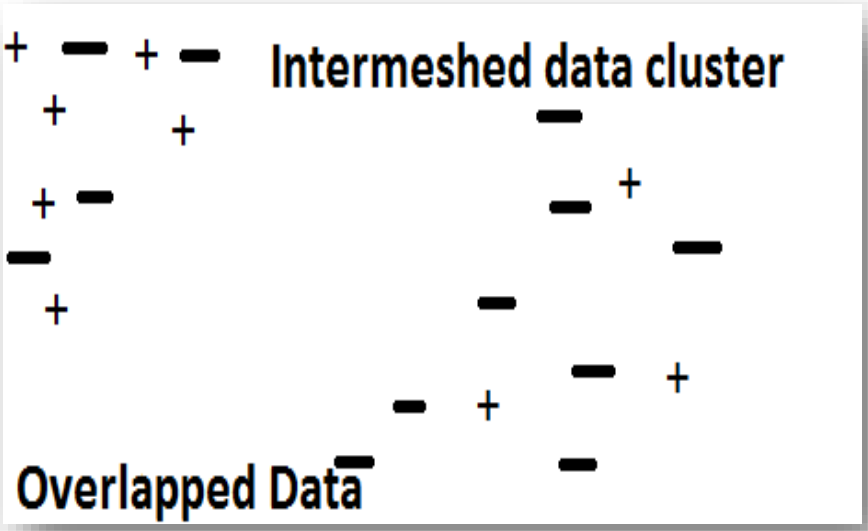


Figure – 2.2: Two intermeshed clusters with overlapping data points

SVMs for binary classification

Consider a binary-classification task with data-points x_i (i=1, 2…m), having corresponding labels $y_i = \pm 1$ and the decision function is:

$$f(x) = \text{sign}(w.x + b)$$

Where . is the inner or scalar product so that $w.x \equiv w^T x$.

From the decision function it is clear that the data is correctly classified if $y_i(w.x_i + b) > 0 \; \forall i$ since $(w.x_i + b)$ should be positive when y_i = + 1, and negative when y_i = - 1.

The decision function is invariant under a positive re-scaling of the argument inside a sign-function. Hence, we implicitly define a scale for (w, b) by setting $w.x + b$ = 1 for the closest point on one side and $w.x + b$ = -1 for the closest

on the other side. The hyperplanes passing through $w.x + b = 1$ and $w.x + b = -1$ are labeled as canonical hyperplanes, and the region between these canonical hyperplanes is called the margin band [15, 61, 77]. This is shown is Figure – 2.3 with H1 and H2 as two canonical hyperplanes.

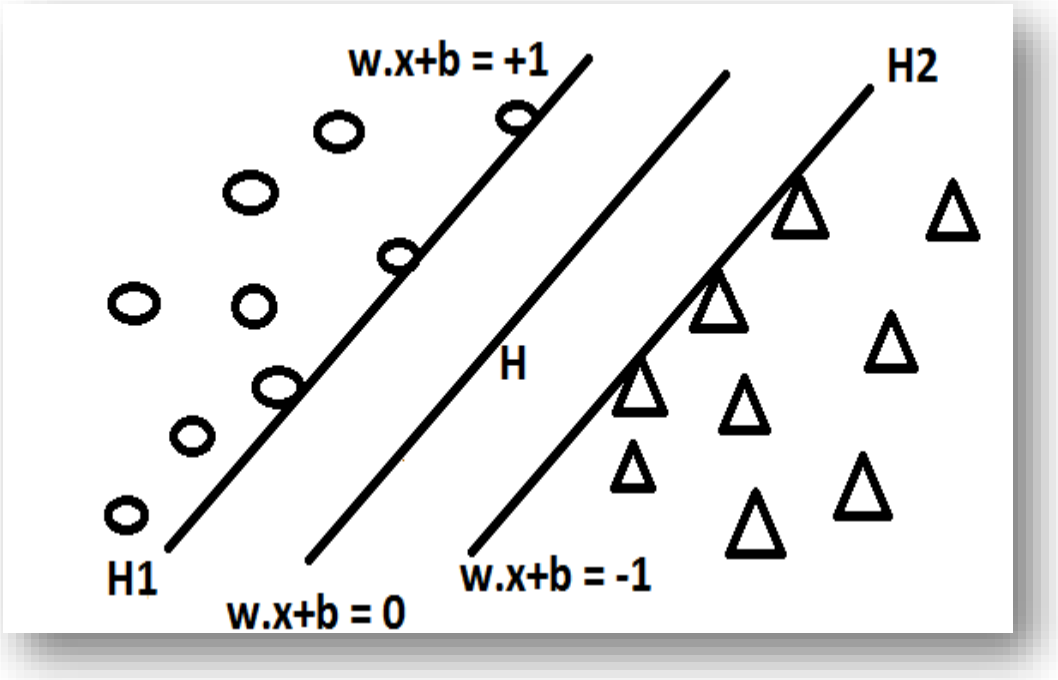


Figure – 2.3: Canonical hyperplanes with margin band

It is observed that the data-points, x_i, only appear inside an inner or scalar product. In order to get an alternative representation of the data, we could therefore map the given data-points into a new space called 'feature space' which has a different dimensionality and may be represented as below:

$$x_i . x_j \rightarrow \emptyset(x_i) . \emptyset(x_j)$$

Where Ø (.) is a mapping function. One major reason for performing this mapping is that the presented data may not be linearly separable in input space as shown in Figure – 2.2. It may be very complex to find a directed hyperplane separating the two classes of data, and the above argument fails. Data which is inseparable in input space can always be separated in a space of high enough dimensionality. However, including a third dimension to the scenario helps in separating the data into two classes as shown in Figure 2.4. In this case, hyperplane is defined in feature space and is non-linear in nature.

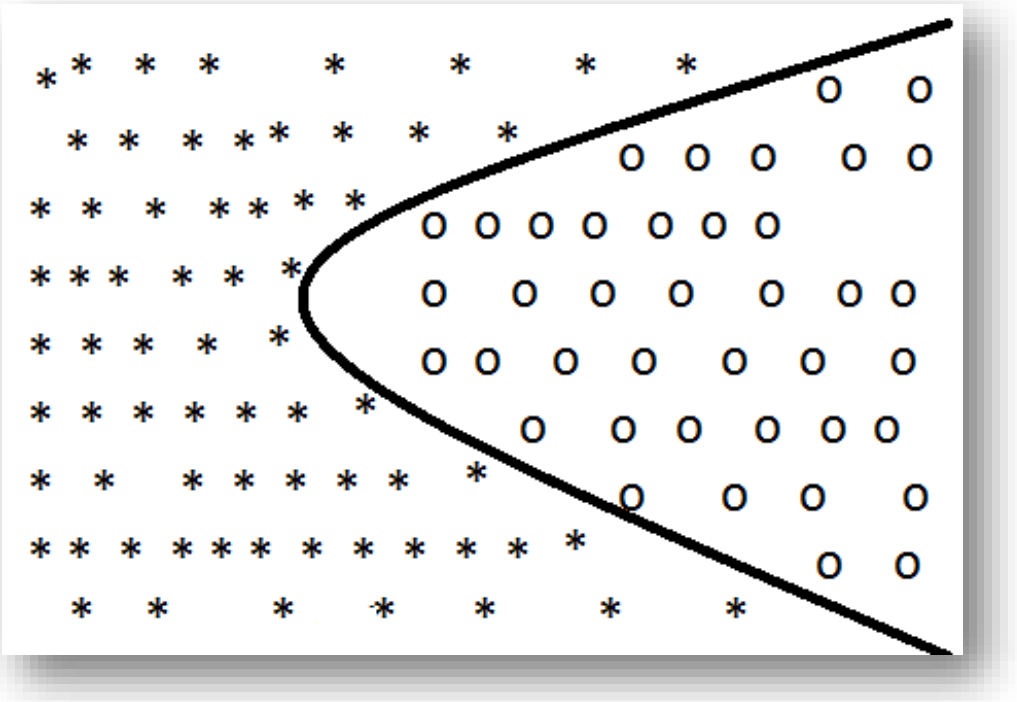


Figure – 2.4: Non-Linear hyperplane in feature space

The function form of the mapping $\emptyset(x_i)$ is not required to be explicitly known since it is implicitly defined by the choice of a new parameter defined as 'kernel' [76, 89].

$$\text{Kernel} \quad K(x_i, x_j) = \emptyset(x_i).\emptyset(x_j)$$

It is defined as the inner or the scalar product in the feature space.

SVMs are mainly used for classification of datasets which may be linearly or non-linearly separable [77]. Suppose a two-class dataset is presented and it is not sure whether the data can be linearly separable or not. We could start with a linear kernel $K(x_i, x_j) = \emptyset(x_i).\emptyset(x_j)$ with no mapping to feature space. If the data is linearly inseparable then we would not get zero training error and the dataset will be misclassified. Though the data may not be separable in input space, it becomes separable in a higher dimensional space by using non-linear kernels. We have many basic non-linear kernels available, for example, polynomial, Radial Basis Function (RBF), Gaussian etc. The introduction of a kernel with its implied mapping to feature space is known as kernel substitution.

Basic non-linear Kernel functions used in SVMs: The two commonly used families of kernels are Polynomial and (RBF) kernels.

Polynomial Kernel function

This type of kernel is applied when we have two types of features which are not linearly separable in 2D space. We need to transform the function into higher dimensional space. Using

a polynomial kernel, these points can be linearly separated in Poly-Dimensional space.

The polynomial kernel is a non-stationary kernel and is well suited for problems where we need to normalize all training data [82].

$$k(x,y) = (\alpha x^T y + C)^d$$

Here, we have α = slope of the hyper-plane; c = constant; d = degree of polynomial

The case of d = 1 points to a Linear Kernel and d = 2 gives us quadratic kernel.

We can re-write above equation in terms of ϕ (x)

$$k(x_{i,} y_j) = \emptyset (x_{i,})^T \emptyset (y_j)$$

A polynomial kernel models feature conjunctions up to the order of the polynomial. Let us take a real world scenario. The requirement is to model occurrences of pairs of words which provide distinctive information about the topic classification, but it is not possible to give the information by the individual words alone. This scenario prompts us to use a quadratic kernel. If distinctive information is provided by the occurrences of triples of words, then we essentially use a cubic kernel and so on.

RBF Kernel

The RBF Kernel function is given by [82]

$$k(x,y) = \exp(-\sigma \|x - y\|^2)$$

Where σ is a positive parameter controlling the radius.

RBF is equivalent to mapping the given data-points into an infinite dimensional Hilbert space where a Hilbert space is an abstract vector space having a similar structure of an inner or scalar product which extends a given dimensional space to spaces with any finite or infinite number of dimensions [15]. For RBF kernel, the maximum value of $\|x - y\|^2$ is n. Therefore, a normalized kernel may be computed as

$$k(x, y) = \exp(-\frac{\sigma}{n} \|x - y\|^2)$$

In the feature space, the norm of every sample determined by the RBF kernel is unique and positive. Hence, the given samples will be mapped onto the surface of the hyper-sphere.

The cosine values, i.e., the values of the RBF kernel function, indicate the similarities between samples as below:

- If the cosine values are close to 1, then these samples are more similar in the feature space.
- If the cosine values are close to 0, then these samples are more dissimilar in the feature space.

Gaussian kernel

The Gaussian kernel is an extended example of RBF kernel [82].

$$k(x, y) = \exp(-\frac{\|x - y\|^2}{2\sigma^2})$$

The adjustable parameter σ (sigma) plays a critical role in the performance of the kernel which must be prudently tuned

to the given situation. If over-estimated, the exponential will behave very nearly linear and in turn, the higher dimensional projection will start to lose its non-linear power. On the other hand, if underestimated, the function will be short of regularization and the decision boundary will be highly sensitive to noisy training data. Elaborated explanation on SVM and its basic kernels can be found in [20, 21, 45, 46, 90].

SVMs are acknowledged by the researchers as a highly effective and efficient classifier vis-à-vis its alternatives available around the board. A few text classification experiments carried out on SVM, TF-IDF, Naïve Bayes, Decision Tree C4.5 and Multi-Layer Perceptron show SVM as a distinct first in this comparison. Also, in another experiment, SVM is found far more superior than ANN (Artificial Neural Network) when compared on same feature data namely Shadow based, Chain code Histogram, Longest Run, and View based features. Multi-class classification with Gaussian kernel SVM gives better results when compared with ANN. SVM has been found to be a better character recognizer because it uses structural risk minimization (SRM) by maximizing margin of separation in the decision function. SVM is compared with HMM (Hidden Markov Model) on various parameters like rejection rate, error rate etc. The SVM presents better stability when the number of training samples is increased. This shows the intrapersonal variability and interpersonal similarity capability of SVM without even possessing previous knowledge. Performance, accuracy and F-measure of SVM can be improved by using TF-BNS (Bi-Normal Separation) instead of TF-IDF. Although SVM is better in overall performance as compared to

KNN and NB, but it is also found that KNN and NB outperform SVM if suitable preprocessing is carried out on the data.

Advantages of SVMs

- SVMs can be very useful for insolvency analysis, in the case of non-regularity in the data. This implies that SVMs are pretty handy when analyzing non-regularly distributed data or data which have an unknown distribution or large number of dimensions.

- Kernels in SVMs provide a huge flexibility during the choice of the threshold criteria separating solvent from non-solvent data-points. This can be done either using linear or non-linear model.

- Kernel implicitly has a non-linear transformation, hence no prior assumption or knowledge of transformation is required which makes the given data linearly separable. In case of Gaussian kernel implementation, if various parameters are properly chosen, SVMs can provide a very good generalization solution. This leads to robust SVM model even if there is a little bias present in the training samples.

- In Neural Networks, we get multiple solutions associated with local minima which lead to not-so-robust unique solution. This issue is addressed in SVMs that deliver a unique solution since the optimization problem is convex in nature.

- Finding similarity between two samples is easy and more reliable when we use SVM as classifier.

- Over-fitting in SVMs is avoided by choosing the maximum margin separating hyperplane from the group of probable solutions that may separate the positive examples from negative ones in the feature space.

- Prediction accuracy is generally very high in SVMs and they are robust to handle noisy data for classification.

Disadvantages of SVMs

- Lack of transparency of derived results is the major drawback of a non-parametric technique like SVMs.

- It acts like a black box oracle and very difficult to comprehend. The way-out for this issue is extraction of association rule in SVM.

- The critical limitation of SVMs lies in the choice of the kernel and its function parameters.

- Time taken during training of data samples poses a big bottleneck during SVM modeling.

- Although, SVMs perform reasonably well during generalization, sometimes they are abysmally slow.

- From practical viewpoint, there is a serious problem with SVMs in terms of high algorithmic complexity and extensive memory requirements of the required quadratic programming while dealing with large-scale classification problem.

- Problem needs to be formulated as 2-class classification. Multi-class classification is still an open research area. It is not easy to incorporate domain knowledge in SVMs.

- Also, over-fitting in SVM is not altogether uncommon because of selection criteria in kernel models. It may lead to performance issues and thus cannot be ignored fully.

Basic Kernels functions for SVMs

One of the most critical problems in SVM modeling is kernel selection for a particular task and dataset. Different kernels for SVM models have been evaluated and compared by many researchers. One comparison shows that while comparing the performance of different kernels for the identification of adult videos with a fixed false alarm of FP = 0.4, it has been experimentally shown that Gaussian kernel performs at the highest level of 97% and linear kernel performs at the lowest level of 76% when true positive (TP) of detection is considered as a parameter. While working on text-independent speaker identification using the TIMIT corpus, a detailed evaluation of various kernel methods is carried out using MATLAB toolkit. Polynomial, RBF and Linear kernels are compared during the experiment. It has been observed experimentally that SVMs trained using polynomial kernel has provided the best performance as compared to other kernels even for large datasets. During face recognition experiment [52], it is found that RBF kernel scores over linear, polynomial and sigmoid kernels. Although SVM and ANN have similar results in binary classification, SVM greatly out-performs ANN in multi-class classification when SVM is used with Gaussian kernel [51]. Multi-word extraction using linear kernel outperforms non-linear kernel during the classifying experiment.

2.5 Semantic Kernel Functions for SVMs

There is not even an iota of doubt about huge success of web during the last couple of decades. WWW is the most useful and adventurous platform for information exchange among global citizens. It is a challenge to extract useful information from this corpus of humongous data. A major area for exploration while learning and extracting knowledge from the unstructured text, is to find similarity / dissimilarity measures among concepts and features. Linkage of similar concepts in a given dataset leads to optimal knowledge discovery. Kernel methods like SVMs are commonly used learning mechanisms for mining unstructured data. These learning methods have two basic components namely kernel machine and kernel function. The learning task is encapsulated by kernel machine and learning hypothesis is imbibed by kernel function. These kernel methods were initially used to evaluate concept similarity based on syntactic information of the given data. But last few years have seen a lot of work being done on mining the text based on semantic information. Semantic based text analysis exploits the principal of knowledge discovery based on the semantic relationships among concepts and features instead of syntactic similarity / dissimilarity.

Our objective in this literature survey is to explore and evaluate various semantic kernel functions being implemented in the context of text classification. We have chosen SVM as kernel method because these classifiers have shown very promising results when compared with Decision Trees, Bayes Nets, KNN and Naïve Bayes to name a few [31]. Availability of linked data on the web has encouraged us to also evaluate the

utility of semantic kernel functions in the context of semantic web, although quite a few kernel functions explored during the survey have been developed independent of semantic web context. Eight commonly used semantic kernel functions are assessed with respect to learning methods, semantic modelling, algorithm / technique used during the learning process and their utility in the context of semantic web.

A kernel function can be viewed as a function that encodes a precise notion of similarity of data items of the input domain, and may simultaneously serve three purposes [10]:

- It provides interface between learning algorithm and data, which is particularly of interest to data items that do not take the traditional form of vectors.
- It can leverage the performance of the algorithm by incorporating a-priori knowledge about problem domain.
- Its evaluation might be computationally valuable compared to an explicit construction of the feature space in terms of system memory and other computation needs.

However, restrictions are applied on how the employed function is chosen to make it a valid kernel, i.e. a positive semi-definite function. Thus it is not a trivial task to construct appropriate positive semi-definite kernels from arbitrary data. However, several closure properties aid the construction of positive definite kernels from known valid kernels [10].

Kernel function maps the original feature space of the data set under consideration into a high-dimensional space, which helps in simplifying the learning task. Such a mapping is not

explicitly performed (kernel trick): the usage of a positive definite kernel function (i.e. a valid kernel) ensures that the embedding into a new space exists, so that the kernel function corresponds to the inner product in this space. In this manner, an efficient algorithm for attribute-value instance spaces can be converted by merely replacing the kernel function into one suitable tree or graph like structured spaces [74].

Most of the text classification systems implement tree or graph type kernels that exploit syntactic structures of the given textual data. These kernel functions do not take into account the semantic relationships between various terms and concepts. In such systems, it is more probable that the documents having related topics but using syntactically different terms are mapped to very distant regions of the feature space. Semantic kernels are particularly very useful in such scenarios where similarity between documents is computed considering semantic relationship between the terms and are subsequently mapped near to each other in the feature space.

In this literature survey, we explore eight semantic kernel functions based on some critical parameters like

- Learning method (either concept based or similarity based)
- Algorithm / technique used during learning process
- Semantic Information they model
- Resources used during implementation process
- Dependency on external or internal dataset
- Learning for or from semantic web

2.5.1 Semantic Smoothing Kernel

This is among the very first methods that suggested text categorization by implementing semantic kernel. The method [81] exploits the information provided by WordNet, a hierarchical semantic database of English words. In general, WordNet provides relationships between words which work as “is-a” relationship for nouns, adverbs and adjectives in the form of hypernym and synonym links. The length of the path between any two words indicates if two terms are semantically close or not.

All documents in the corpus are represented statistically by defining a-priori semantic proximity between two words in the interval of [0, 1]. If two words are semantically close, the relationship interval between them is defined as’1’ else it is ‘0’. Once the training corpus has been prepared with the building of index of most relevant terms, a proximity (symmetric) matrix ‘P’ is formed that reflects semantic relations between the index terms. Its dimension is thus the size of the index. It performs semantic smoothing of the vectorial data by defining terms as a-priori strongly related if they are semantically close to each other and a-priori isolated if they are semantically not close to each other.

For two vectors x and y, ordinary Euclidean metric is transformed into the following measurement where positive semi-definite matrix ‘S’ is defined as $S = P^2$

$$||\mathrm{P}\,(\mathrm{x}-\mathrm{y})||^2 = (\mathrm{x}-\mathrm{y})^{\mathrm{T}}\,\mathrm{P}.\mathrm{P}(\mathrm{x}-\mathrm{y})$$

$$||\mathrm{P}\,(\mathrm{x}-\mathrm{y})||^2 = (\mathrm{x}-\mathrm{y})^{\mathrm{T}}\,\mathrm{S}(\mathrm{x}-\mathrm{y})$$

This measurement can be incorporated to the definitions of a kernel in SVM. To implement this kernel function ‘K’, RBF is used with above matrix as an argument of exponent.

$$K(x, y) = \exp(-\sigma \,||\, x - y \,||^2)$$

After semantically smoothing the vectors, we get

$$K(x, y) = \exp(-\sigma \,||\, (x - y)^T S (x - y) \,||^2)$$

This matrix is incorporated to the definition of a kernel in SVM.

This simple but efficient semantic smoothing kernel can be seen as a-priori mapping of feature vectors to a semantically more suitable space. It provides a good case to apply powerful learning algorithm like SVMs as soon as a positive matrix ‘S’ can be defined over the data.

The kernel is implemented using 20-NewsGroups dataset which has 20,000 UseNet articles partitioned in 20 thematic categories. Each theme contains 1000 articles. SVM and KNN are being used and compared as kernel methods. Multi-class categorization problem is divided into 20 separate bi-class problems where total articles in that specific category are considered as positive and an equal number of articles randomly chosen among all the other classes as negative examples. Size of training set is 2/3rd of the corpus. Test is performed on 150, 200, 250 and 300 words which are selected by the highest mutual information criteria. Experimentally, it is found that increasing the number of indexing words

beyond 200 doesn't improve the performance. Results shows that introduction of the new a-priori metric increases the classification rate. Also, kernel method SVM performs much better than KNN. The overall complexity of the calculation needed to generate the semantic proximity metric in the kernel method increases with the increase of indexing terms. Semantic proximity metric in the kernel increases the number of support vectors thus extraction of support data cannot be very selective. A few more variants of semantic smoothing kernel are suggested by [12, 58].

2.5.2 Latent Semantic Kernel (LSK)

Semantic Smoothing Kernel uses an external semantic network (WordNet) to explicitly compute the similarity levels between the terms. There is another way to find out the semantic similarity among the terms. It can be done by using statistical way of extracting term-term co-relations derived from the corpus itself instead of depending on external reference like WordNet [21]. This approach is the extension of semantic smoothing kernel technique using Latent Semantic Indexing (LSI) [30]. The semantic similarity between the terms may be inferred by simply analyzing their co-occurrence patterns. The terms are considered as related if they co-occur often in the same documents. Singular Value Decomposition (SVD) is used to extract the co-occurrence information statistically (for more information on SVD, please refer [40]. Once the dimension of new feature space is fixed, its computation is equivalent to solving a convex optimization problem of eigenvalue decomposition. The latent semantic index metric is made a part of kernel which in turn may be used with

kernel methods like SVM for the text categorization. This kernel is termed as Latent Semantic Kernel (LSK). It explicitly computes the similarity level between terms in a given semantic network, and defines a new metric in the feature space [75].

The main drawback of this kernel is the difficulty to process large training sets which are more than a few thousand examples. It is because of the fact that this kernel observes computational complexity while performing eigenvalue decomposition on the kernel matrix. This may be addressed by considering an iterative approximation method that is equivalent to projecting onto the first dimension derived from a Gram-Schmidt orthogonalisation of the data. The semantic kernel from this approximation method is termed as Gram-Schmidt Kernel (GSK) which is experimentally shown as more effective than LSK

Experimental results on text and non-text data showcase the usefulness and general applicability of LSK in different conditions. Reuters-21578 and Medline-1033 datasets are used for the text categorization while Ionosphere dataset from the UCI repository is used for the non-text data. SVM is used as the kernel method during all experiments with 10-fold cross validation. Consistent higher values of F-measure in case of GSK demonstrate that GSK is a very effective approximation strategy for LSK. Ionosphere data set from the UCI repository is considered for non-text data experiment which contains 34 features and 315 points. It is found experimentally that classification error for semantic kernel LSK is minimum when compared with polynomial kernel. A generalized GSK is also

implemented with Reuter's documents transformed into a new feature space using SVM-Light. The micro-averaged F-measure for a SVM with generalized GSK is 0.822 whereas the micro-averaged F-measure for an SVM with linear kernel is 0.854. These comparisons reflect that the performance of the proposed technique is comparable to the baseline method and also highlights the generalized GSK as a practical approximation of LSK.

2.5.3 Semantic WordNet Based Kernel

This kernel also uses a term similarity measure based on the WordNet (WN) hierarchy. It introduces the semantic lexical knowledge contained in the WN hierarchy in a supervised text classification task [6].

Let there be a set of documents D, with terms in the documents being denoted by $t \in D$ and terms as nouns $t' \in V$, where V is vocabulary of WN nouns. The terms in the documents are represented through a set of all pairs in the vocabulary with $< t, t' > \in V$. We get a well-defined space which supports the similarity between terms of different surfaces based on external knowledge and thus we avoid defining terms of sense clusters explicitly.

This kernel exploits the theory of Concept density (CD) [3]. CD is a flexible semantic similarity which is dependent on the generalization of word sense without referring to any fixed level of the hierarchy. CD is primarily represented as a metric that incorporates topological structure of WN and may be applied to two or more words.

If u_1, u_2 are two words, S_1, S_2 are senses of u_1, u_2 represented in WN, and h being the depth of semantic tree to cover two senses, then

$$CD\,(u_1, u_2) = \begin{cases} 0\ if\ S_1 \cap S_2 = \emptyset \\ \max if\ S_1 \cap S_2 = h \end{cases}$$

CD models the semantic distance as a density of the generalization $\in S_1 \cap S_2$. Semantic WN based kernel is defined with the help of CD function mentioned above.

Given two documents $d_1, d_2 \in D$ (the document set), their similarity may be defined as

$$K\,(d_1, d_2) = \sum_{w_1 \in d_1, w_2 \in d_2} (\lambda_1\ , \lambda_2\ ) \times \sigma\,(w_1, w_2\ )$$

Where λ_1 and λ_2 are the weights of the words (features) w_1, w_2 in the documents d_1, d_2 respectively and σ is a term similarity function CD (as calculated above).

This kernel is experimentally implemented using SVM as the kernel method and duly compared with vector space model (VSM) to highlight its better performance. 20-NewsGroups and Reuters-21578 corpus are used as datasets. Given the high computational complexity of semantic kernel, 8 categories each from 20-NewsGroups and Reuters-21578 corpus are selected. The experiment confirms better performance of semantic kernel when compared with "Bag of Words" kernel on a small size of training set (Improvement in micro-average Performance with 24 documents is 12% but the same is 2% when we have 160 documents).

In semantic WordNet kernel, relatedness of each term occurring in the first document is computed against all terms in the second document. This means that 'similarity' instead of 'identity' of terms is the key in this technique.

There are three differences when we compare WordNet Semantic Kernel with Semantic Smoothing kernel.

- Term proximity in semantic smoothing kernel does not fully capture the WordNet topological information. Equidistant terms receive same similarity irrespective of their generalization level.

- In Semantic Smoothing kernel, weighting schemes, the RBF kernel and the proximity metric are used collectively. This offers a much less clear interpretation. But in case of CD based similarity kernel, combination of lexicalized and semantic information provides better understanding.

- No. of features considered by CD based WordNet kernel is much larger than semantic smoothing kernel.

The difference between LSK kernel and semantic WordNet kernel is primarily the use of different source of prior knowledge.

2.5.4 Kernel with Implicit Superconcept Expansions

Semantic smoothing kernel encoded the knowledge in the form of semantic network explicitly from an external source i.e. WordNet. LSK derived the same knowledge implicitly from statistics about the co-occurrence of terms. In these

approaches, index terms of feature space cannot be considered as mutually orthogonal dimensions. These can be regarded as dimensions with varying degrees of semantic similarity. This leads to a typical assumption that we can detect stable knowledge patterns even in poor representation as long as sufficient training data is available. We need to investigate the overall performance of these kernels in the scenario when training data is scarce or the representation of individual instances is hampered by extreme sparseness.

A new kernel [12] tries to detect knowledge patterns from a given dataset in case of insufficient training data. It exploits the similarity between the base concepts and their superconcepts. In a given semantic network having two concepts c_1, c_2, the relation $\text{super}(c_1, c_2)$ indicates that c_1 is superconcept of c_2 . Distance 'd' of two concepts c_1, c_2 may be referred to as the number of superconcept edges between c_1, c_2 that can be easily computed using the Flyod-Warshall algorithm [43, 65]. The notion of depth (dep) of a concept relates to a tree-like structure of the semantic network having a unique root element. For acyclic graph, a root element becomes superconcept of all concept nodes that are not equipped with outgoing superconcept edges. The depth of a concept is then defined as the distance of the concept to the root. Lowest super ordinate (*lso*) of two concepts refers to the concept with maximal depth that subsumes them both.

The inverted path length (IPL) is the simplest way for computing the semantic similarity between two concepts:

$$\text{sim}_{\text{IPL}}(c_1, c_2) = \frac{1}{(1 + d(c_1, c_2))^{\alpha}}$$

Here α is the rate of decay.

This is also used by [81] for defining semantic smoothing kernel. Although simple and intuitive in approach, IPL does not provide required accuracy. This is because it does not comply with the intuition that concepts closer to the root of the semantic network should have a higher distance compared to concepts far away. Therefore, [12] suggests a modified semantic kernel termed as semantic superconcept kernel that relates distinct but similar features during evaluations simply by embedding the knowledge about the topological relations of the semantic network in kernel function. The definition of semantic smoothing kernel implies that 'S' must be a positive semi-definite matrix which can be decomposed as ($P.P^T$) thus revealing the underlying feature mapping. The requirement of 'S' being positive semi-definite may not be typically ensured in general scenarios. Enforcement of positive definition of 'S' provides a way to avoid indefinite similarity matrices. While this approach ensures the kernel validity, the interpretation of smoothing kernel is less clear. Basically, it maps each concept to a number of related concepts and the shared weight of these provides the overall similarity between the two terms. In this case, 'P' is setup such that it provides a mapping into the space of all possible superconcepts of the input concepts. Each row in 'P' corresponds to the vector representation of the concepts of the input space by means of their respective superconcepts. The similarity of two concepts in the resulting smoothing matrix 'S' is thus the dot product of the vectors of their respective superconcepts. It is assumed here that two concepts

are more similar if they share a large number of superconcepts as opposed to sharing only a few superconcepts.

Different weighting schemes are used to represent superconcepts in 'P'. First being considered is the weight of superconcept c_j during the vector representation of a concept c_i is influenced by its distance from c_i. Secondly; it is also influenced by its overall depth in the semantic network. The Superconcept Kernel kS for two concepts c_i ; $c_j \in C$ is given by

$$kS\left(c_i, c_j\right) = < SC\left(c_i\right) . SC\left(c_j\right) >$$

$SC(.)$ is a function that maps each concept to a real vector R. Its dimensions correspond to superconcepts present in the employed semantic network and respective entries are determined by a particular weighting scheme. The concept kernel $kS\left(c_i, c_j\right)$ can be used directly in conjunction with the standard linear kernel by means of a simple Semantic Smoothing Kernel.

Experimental evaluation is done on Reuters-21578 and the TREC QA datasets using SVM-Light. WordNet is used as the underlying semantic network. Two simplifying assumptions are made – first for bag of words representation, term proximity matrix is used and secondly for Reuters-21578 dataset, terms having frequency more than five are considered during experiment. F-measure on Reuter's dataset provides a consistent improvement of 15.5% when semantic superconcept kernel is used vis-à-vis linear kernel, whereas in case of TREC QA dataset, maximum 9.32% is observed. Overall, this approach offers a consistent improvement in performance in

those scenarios where little training data is available or the feature representations are extremely sparse.

2.5.5 ALC KERNEL

Description Logics (DLs) have been adopted as a core technology for ontology languages like OWL [5]. Description Logics have the ability to represent other kinds of relationships that can hold between concepts, beyond "is-a" relationship. A description logic Knowledge base $\mathrm{K} = < T, A >$ comprises of two components – a "TBox" and an "ABox". TBox contains intensional knowledge, or general knowledge about the problem domain in the form of terminology (hence called TBox). ABox contains extensional knowledge, which is specific to a particular problem in the form of assertional knowledge (therefore the term "ABox"). Knowledge in description logics and subsequently description of ontologies use constructs that have semantics given in predicate logic.

AL (Attribute Language) logic is part of description logics that defines these constructs with the help of practically usable vocabulary. Some of the syntax used in AL are atomic concept, atomic role, top (most general) concept, bottom (most specific) concept, atomic negation, intersection to name a few. More details about AL-Concepts (ALC) can be found in [73].

Descriptions are inductively defined with the help of primitive concept names (N_c) and primitive roles(N_R). Interpretation function (I) is a function which maps each primitive concept name with appropriate primitive role with the help of concept conjunction and disjunction relationships.

Disjunctive ALC kernel may be defined as a kernel which primarily computes the similarity between given disjunctive descriptions as the sum of the cross-similarities between any couple of disjuncts from either description [32]. Conjunctive kernel computes the similarity between two input descriptions by distinguishing given primitive concepts into two categories namely those referred in the value restrictions and those referred in the existential restrictions. These similarity values are multiplied, thereby reflecting the fact that all the restrictions have to be satisfied at a conjunctive level. The similarity between primitive concepts is measured in terms of the intersection of their extension.

Concept retrieval may be done by means of SVM that use ALC kernel function. The basic problem is to map descriptions of TBox to the concepts of ABox which is primarily a multi-class problem. This multi-class problem is decomposed into smaller binary classification problem which points to classification of concepts into two assertions i.e. existence or non-existence categories. Also, this kernel function considers the general classification problem as Open World Assumption (OWA) instead of Closed World problem (CWP). This implies that under OWA, we do not consider the absence of information for a given concept as 'negative' which is usually the case on CWP. This provides a very interesting consideration of another set apart from binary set of values. Therefore, the evaluation will be done using the set $\{+1, -1, 0\}$. The consideration of third category actually helps the system to improve concept retrieval service which may be the base for 'Inductive Learning'.

The ALC semantic kernel is experimentally implemented with SVM as kernel method. The setup has exploited nine ontologies from the protégé library. The classification method is applied to all the individuals in each ontology just to assess if these individuals are instances of the concepts in the ontology. For each concept in the ontology, match rate (avg. 0.84), omission error rate (avg. 0.089) and commission error rate (avg. 0.025) are computed. Match rate is equal to the number of cases of individuals that exactly got the same classification with respect to the overall number of individuals. Omission error rate provides the amount of unlabeled individuals that are not classified as concepts. Commission error rate computes the amount of individuals labeled as instances of a concept while they logically belong to other concepts. Here omission error rate helps in computing inductive learning rate of the system.

ALC kernel function provides a thumping performance guaranteeing almost null commission error. Interestingly, it has the ability to induce new knowledge also. The performance of the kernel function improves when we improve the concepts available in ontologies. The performance of the classifier may be improved with the increase of the number of individuals populating the considered ontology that have to be preferable homogeneously spread with respect to the concept in the ontology. The key weakness of the kernel function approach is on its scalability towards more complex description logics.

2.5.6 Description Logic Kernel

Most of the semantic kernels discussed till now address the problem of learning ‘for’ the semantic web. But in today’s

time we have very well populated semantic web available which is supported by huge ontologies covering a lot of domains. There is a semantic kernel by the name DL (Description Logic) kernel that helps in learning 'from' the semantic web [25, 26]. It encodes a notion of similarity of individuals, by exploiting only semantic aspects of the reference representation. Most of the kernel functions discussed above are deductive in nature which perform reasonably poor when the dataset is from heterogeneous and distributed sources. Description logics kernel primarily works on inductive learning principle.

Consider a triple $< N_c, N_R, N_I >$ made up respectively, of a set of concept names N_c, a set of role names N_R, and a set of individual names N_I. An interpretation "I" maps concepts with descriptions. Description logic language provides the set of rules for building more complex concept descriptions based on these building blocks by extending appropriate interpretations among specific constructs. These concepts are similar to what we have discussed in ALC kernel, but the limitation of ALC kernel is its dependency on description logic language. Description logic kernel addresses this problem by applying the kernel directly to individuals, based on their inductive distance measures. Two individuals are similar w.r.t. to a given concept if they exhibit the same behavior i.e. both are instances of the concept or its negation. Conversely, the minimal similarity holds when they belong to opposite concepts.

The performance of description logics kernel is not only comparable to that of a standard reasoner, but the classifier

is also capable of inducing new knowledge, which is not logically derivable. Particularly, an increase in prediction accuracy is observed when the instances are consistently spread, as estimated from statistical methods. This realized classifier can be exploited for predicting / suggesting missing information about individuals, thus completing large ontologies.

Also, description logic kernel is compared with ALC Kernel [32] and experimentally it is shown that it improves both match rate (0.966) and omission rate (0.0275) w.r.t. ALC kernel (match rate 0.923 and omission rate 0.0502). Consequently, a decrease of the induction rate is observed (0.026 for ALC and 0.0065 for Description Logic kernel). The commission rate for the ALC kernel is almost null as that for the description logic Kernel.

2.5.7 Syntactic Semantic Tree Kernel (Syntactic-STK)

Most of the kernel functions explored till now either use only semantic background knowledge or extract relations from the corpus itself. But a new family of kernels called SSTK is proposed [11] that simultaneously incorporates linguistic structures (e.g. syntactic dependencies) and semantic background knowledge (e.g. term similarity based on WordNet) in a single algorithm. Tree kernel encodes the syntactic structure in the form of parse tree and semantic smoothing kernel implements term similarity based on the background semantic knowledge coming from WordNet.

The main rationale behind the Tree Kernel is to characterize trees in terms of their sub-structures [18]. The kernel

function then counts the number of tree sub-parts common to both argument trees. A tree is defined as a connected direction graph with no cycles. Structures are parse trees; each node along with its children nodes is associated with the execution of a given grammar production rule. The labels of the leaf nodes of the parse trees correspond to terms, whereas the pre-terminal symbols are the parents of leaves.

The Semantic Smoothing Kernel for two term vectors is given by a square matrix which is symmetric in nature. The entries in this matrix represent the kernel evaluations between the concepts that encode the evaluations of the disambiguation function. The matrix maps concept dimensions to term dimensions that constitute the input space.

SSTK class of kernels is primarily driven by counting all compatible tree fragments of two chosen parse trees weighted by their joint terminology. It exploits linguistic structure and background knowledge about the semantic dependencies of terms at the same time. More precisely, this kernel uses semantic smoothing to improve the matching of tree fragments containing terminal nodes.

Question Classification [51] is one of the most critical application of this kernel. It aims at detecting the right type of a question from the user query. For example, if a user asks to locate a person or an organization, this question classification system must extract the right question before returning the correct answer. A major challenge of Question Classification compared to standard Text Classification settings is that questions typically contain only extremely few words which make this setting a typical victim of data

sparseness. SSTK improves the state-of-the-art question classification, which makes it a prototype of a possible future full-fledged natural language kernel. The newly proposed Semantic Syntactic Tree Kernels outperform the conventional linear / semantic kernels as well as tree kernels, improving the state of the art in question classification.

2.5.8 Shallow Semantic Tree Kernel (Shallow-STK)

Complexity during the implementation of QA (Question-Answer) system is very high because capturing a correct relationship between question and answer requires expensive probabilistic models. Such models suffer from high sensitiveness to irrelevant features and process errors. Shallow semantic information in the form of predicate argument structures (PASs) improves the automatic detection of correct answers to a target question. Shallow semantic representations, having more compact information, prevents the sparseness of deep structural approaches and the weakness of bag of words (BOW) models [60].

Encoding semantic information represented by means of tree structures in a learning algorithm is problematic. Using all sub-structures as features is a huge computationally complex task. Tree kernel is not suitable for PAS as it creates constraints while considering nodes and sub-nodes as whole or none.

Let there be two trees $T_1 \, and \, T_2$, with $\{f_1, f_2, \ldots\} = \mathcal{F}$ be the set of fragments (sub-structures). $I_i(n) = 1 \, if \, f_i$ is rooted at node $n$or $= 0$ otherwise.

The Tree Kernel may be defined as

$$TK(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)$$

Where N_{T_1} is the set of nodes in T_1 and N_{T_2} is the set of nodes in T_2 .

$$\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} I_i(n_1) I_i(n_2)$$

This is equal to the number of common fragments rooted in nodes n_1 *and* n_2.

$$\Delta(n_1, n_2) = \begin{matrix} 0 \; if \; n_1, n_2 \text{ are different} \\ 1 \; if \; n_1, n_2 \text{ are same} \end{matrix}$$

This implies that the two evaluated nodes have to be identical to allow the match to happen. If we make two changes to this Tree Kernel, one being adding SLOT nodes in PAS to accommodate argument labels in a specific order and second being generating a new $\Delta(n_1, n_2)$ that would generate many matches with SLOTs filled with the null labels, the resultant kernel is termed as Shallow-STK. The SLOT nodes are used in such a way that the TK function can generate one or more alike fragments. The new Δ function counts the number of all possible k-ary relations derivable from a set of k-arguments in PAS.

The experiment is focused on question classification and answer re-ranking for web based QA system. A benchmark dataset from TREC is used for the evaluation. During Question classification phase, accuracy of Shallow-STK is 88.8% as compared to 90.4% for Parse Tree (PT) set-up while during

answer classification, Shallow-STK shows 4% better Mean Reciprocal Rank (MRR) values as compared to PT. Experiment results suggest that syntactic information helps better during question classification whereas semantic information contained in PAS gives promising results in answer classification.

If we relax the constraints on the production rule of Shallow STK, we get a more generic tree kernel called Partial Tree Kernel (PTK) which has a simpler $\Delta(n_1, n_2)$. PTK is able to extract a richer set of features which take gaps into account. Moreover, SLOTs are not required in PTK to extract fragments containing argument subsets which results in increased matching accuracy.

Table – 2.1 provides a comparison of various semantic kernel functions suggested in the field of text classification.

Parameter	Entity	[18]	[60]	[32]	[21]	[9]	[12]	[26]	[11]
Kernel Function	Semantic Smoothing Kernel	√					√		
	Dependency Tree Kernels		√						√
	Superconcept Kernel						√		
	ALC Kernel			√					
	Latent Semantic Kernel (LSK)				√				
	Semantic WordNet-based Kernel					√			
	Description Logic (DL) Kernel							√	
	Syntactic-STK								√

	Shallow-STK		√						
Learning Method	Concept Based	√		√					
	Similarity Based		√		√	√	√	√	√
Efficacy for semantic web	Learning FOR semantic web	√	√		√	√	√		√
	Learning FROM semantic web			√				√	
Dependency	External Semantic Datasets	√	√			√			
	External Ontologies			√				√	
	Internal				√		√		√
Algorithm / Technique	Symmetric Proximity Matrix	√							√
	Tree dependency		√						√
	Superconcepts		√				√		
	Inductive learning of concepts			√				√	
	Latent Semantic Indexing				√				
	Singular Value Decomposition				√				
	Concept Density					√			
Kernel Method	SVM	√	√	√	√	√	√	√	√
Data Set used	20-NewsGroups	√				√			
	Reuter-21578				√	√	√		
	Medline-1033				√				
	TREC		√				√		√
	Nine ontologies from Protégé			√				√	

Table – 2.1: Semantic Kernel Functions compared on various parameters

As we can observe from Table – 2.1, most of the researchers have used Reuters-21578, 20-NewsGroups and TREC datasets during their kernel implementation exercise. Also, only two approaches have used ontologies as their dataset for experimentation.

This literature survey has investigated the usefulness of semantic kernel functions in scenarios of text classification where similarity between terms, concepts and documents is

computed considering semantic relationship between them. Eight commonly used semantic kernel functions are evaluated and compared with each other based on their learning methods, algorithm / technique being used during implementation and other factors like dependency on external datasets during learning and evaluation process. It is also explored if a particular semantic kernel learns ‘for’ or ‘from’ semantic web.

The most important aspect of semantic kernels is a-priori knowledge they exploit while classifying the text. Feature vectors are mapped to semantically more suitable space which significantly improves the performance of text classification. Semantic smoothing kernel is the first step in this direction. It offers a framework to deal with real life problems where we have a-priori domain knowledge available. This kernel is among the very first semantic kernels that propose a new perspective into text categorization when documents are encoded as proximity data rather than numerical vectors. It uses WordNet, a semantic network, to obtain term-similarity information.

Another variant approach that uses WordNet as underlying semantic network exploits “implicit superconcept expansions” while doing the text classification. This kernel is typically useful when the training data is scarce. This finds an application when we implement question classification. Questions normally contain very few words which leads to data sparseness. This challenge is addressed by this semantic kernel which expresses the semantic similarity of term features by means of the shared superconcepts.

These two kernels i.e. semantic smoothing kernel and kernel with implicit superconcept expansions rely on external a-priori semantic knowledge for classification. Another approach namely LSK infers the semantic similarities directly from the corpus itself by using co-occurrence analysis. It uses LSI that captures the semantic relations between the terms and insert them into the similarity measure between two documents. LSK projects the data into a subspace by performing an equivalent projection onto the first eigenvectors of the kernel matrix. Hence, it is possible to apply this technique to any kernel-defined feature space irrespective of its original dimensionality. The design also demonstrate the efficacy of LSK approach on both text as well as non-text data. But for large imbalanced datasets, generally encountered in text classification, it is very complex to obtain good performance. This is because of the fact that number of dimensions grow quite large before relevant features are drawn from the small number of positive documents. GSK helps in addressing this problem by biasing the feature selection in favor of positive documents thereby greatly reducing the number of dimensions required to create an effective feature space.

There is another set of kernels namely ALC and description logic kernels that have the ability to induce new knowledge which is not logically derivable. These kernels are mainly useful in a multi-relational learning environment. They can be exploited for predicting / suggesting missing information about individuals, which in turn, helps in completing large ontologies. This is a very good innovative direction in the context of semantic web since two approaches namely

statistical analysis and symbolic representation work collectively to do the classification. But the major drawback of these two kernels is their inability to scale-up towards more complex description logics. This is because of the fact that the gap between syntactic structure semantics of the descriptions becomes more evident while computing “most specific concept” (MSC) approximations.

Syntactic-STK puts syntactic tree kernel and semantic smoothing kernel together to offer a state-of-the-art question answer classification solution. Experimental results reflect that the SSTK kernel outperforms tree kernel as well as semantic smoothing kernel. It would be interesting to evaluate this kernel along with Latent Semantic Indexing (LSI). Shallow-STK suggests new tree structures based on shallow semantics encoded in PAS. Syntactic information helps in question / answer classification and shallow semantics gives remarkable contribution when a reliable set of PAS can be extracted, e.g. from answers. This is entirely a different approach to question answer classification that introduced new structures to represent textual information in three question answering tasks: question classification, answer classification and answer re-ranking.

The whole evaluation brings an important aspect of using a-priori semantic knowledge in the background. Also, in some cases, syntactic kernel functions also play a critical role in the classification exercise along with semantic kernel functions. Table – 2.2 shows the advantages, limitations and business applications of these semantic kernel functions.

Kernel Function	Advantages	Limitations	Applications
Semantic Smoothing Kernel	• Very simple to implement • Good in improving the performance of classification	• Increased number of support vectors handled by semantic proximity matrix adds to the complexity and thus the extraction of support data cannot be very selective.	• All real life scenarios where domain knowledge is available a-priori.
Latent Semantic Kernel (LSK)	• Possible to apply this kernel irrespective of data original dimensionality. • Can handle both text and non-text data.	• Complexity in performing eigenvalues decomposition. • Large imbalance datasets during text classification poses a threat to good performance.	• Non-text classification e.g. Radar data, satellite data, Ionosphere data classification. • Classification of those datasets where a-priori knowledge is not available easily e.g. classification of medical documents.
Semantic smoothing kernel with implicit	• Performance is consistently good in those cases where	• A scope of further improvement as it does not use a	• Question Classification

Superconcept expansions	little training data is available or the feature representations are extremely sparse.	decent word sense disambiguation step	
Semantic WordNet-based Kernel	• In poor training data conditions, the prior knowledge of WordNet can be effectively used to improve the Text Classification accuracy. • Provides a space which supports the similarity between terms of different surface forms based on external knowledge. • Helps in shunning the need of explicitly defining the term or sense clusters which introduce noise.	• A scope of improvement for the overall efficiency by exploring feature selection methods over the semantic kernel. • The extension of the semantic similarity by a general (i.e. non binary) application of the conceptual density model.	• All real life scenarios where domain knowledge is available a-priori.

ALC Kernel	• Primarily focus on learning from the semantic web. • Pre-processing efforts can be avoided when ALC kernel is used. • Guarantees almost null commission error. • Ability to induce new knowledge.	• Dependency on description logics. • Computationally expensive approximation of the MSC. • Scalability towards more complex description logics.	• Semantic Web • Ontology enrichment • Completing large ontologies by predicting / suggesting missing information about individuals.
Description Logic (DL) Kernel	• Primarily focus on learning from the semantic web. • Improves both match rate and omission rate with respect to the ALC kernel. • Noise tolerant.	• Match rate increases with the increase of the number of individuals in the considered ontology which reflects a conservative behavior of the kernel and may be addressed by using suitable threshold.	• Semantic Web • Ontology enrichment • Completing large ontologies by predicting / suggesting missing information about individuals.
Syntactic-STK	• Uses semantic smoothing to improve the matching of tree	• Can only work on constituency trees and not on	• Question Classification.

	fragments containing terminal nodes, thus providing improved performance.	dependency trees. • Only complete matching of the structure of sub-trees is allowed: there is absolutely no flexibility. • A scope of improvement when it comes to explore different syntactic / semantic structures	
Shallow-STK	• Faster and more accurate than those previously proposed semantic tree kernels	• Computational expensive for real world applications	• Question Classification.

Table – 2.2: Advantages and Disadvantages of Semantic Kernel Functions

The exercise also reflects that in order to gain reasonable performance for semantically hosted data, we must fine-tune statistical learning algorithms like SVMs along with semantic kernels so as to gain the acceptable reasoned semantic web. It is very much critical to enrich ontologies not only for the semantic web but also from the semantic web. This can be achieved either by using ALC or description logic semantic kernel respectively.

In order to leverage the semantic web, it is most important to have a well-suited ontology set-up in place and avoid asking the semantic web to do more that it can do at this point. Ontology enrichment will be quite reasonable expectation at this point which may be explored by applying inductive learning methods for learning probabilistic ontologies. It will help in very low error rates, along with enhanced capability to induce new knowledge from semantic web that is not logically derivable [29]. Also, it will be quite interesting to explore these kernel functions on live web data which may test their actual utility in real business scenarios.

Concluding remarks of semantic kernel function survey:

The domain of text classification has witnessed a gradual transition over the last few years. After using syntactic based algorithms for many years, this domain is slowly embracing systems that use a-priori semantic knowledge to classify the unstructured data available in abundance over web. Kernel functions being the core of such systems have also matured enough to exploit this semantic knowledge. Through this survey, we have evaluated and compared eight commonly used semantic kernel functions on various parameters like learning method, algorithm, dataset dependency and efficacy for semantic web.

Semantic kernel functions find their utility in many areas of text classification. Semantic Smoothing kernel using semantic knowledge from WordNet finds its application in almost all field of real life where domain knowledge is available a-priori. This gives us a quick solution in

scenarios where we have ontologies available and we want to do some quick text analysis. An efficient and easy to implement kernel function, semantic smoothing kernel is one of the most used semantic kernel function across the domain of text classification. Many variants of this kernel have been implemented to serve specific industry problems of textual analysis. Quite a few kernel functions have been implemented for scenarios where we have very less training data like question answer classification. Semantic smoothing kernel with implicit superconcept expansions, Syntactic-STK and Shallow-STK are three kernel functions discussed in this paper primarily address Q&A classification. Non-Text data is also one of the prime area where semantic kernel functions find their application. Latent Semantic Kernel offers a solution for classification of non-text data like images, radar and satellite data. This kernel also works well when we don't have prior domain knowledge available with us e.g. medical document classification. In current times, ontology enrichment and completing large ontologies are two sub-areas of text classification explored in great depth. ALC and description logic kernels find their utility in these fields where they learn "from" semantic web.

2.6 Application of SVMs in Ontology Learning and Information Extraction

Ontologies are schemas of metadata which provides a controlled vocabulary of terms and concepts, with an explicit defined relationship [1]. It is a formal explicit specification of a "shared conceptualization". Ontologies are "semantic containers" which can help text understanding and

automatic processing of text documents available on the web [41].

In an ideal scenario, if the web is equipped with ontology marked documents, it will be an optimal case to get all queries correctly answered by web search engines as if a human being is answering the queries. But SW is yet to achieve the status which it should have on World Wide Web because web documents are still not fully tagged with relevant Ontologies [41]. A strong need is felt to strengthen the processes and algorithms to automate ontology learning so as to remove this bottleneck.

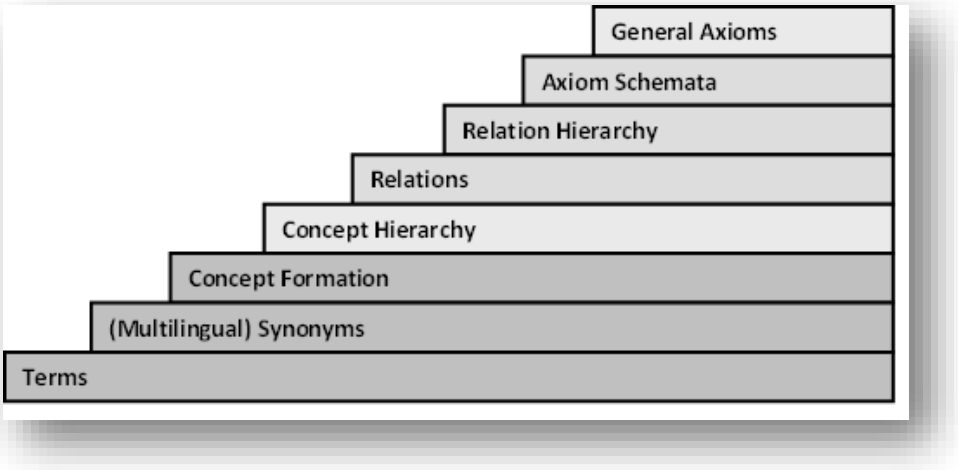


Figure – 2.5: Ontology learning layer cake

Ontology learning can be modeled as lexical entailment problem where entailed words are titled as fine grained classes and named entities belonging to the coarse-grained ontological type are entailing words. Ontology Learning Layer Cake which is highly influenced by Tim Berners-Lee's SW layer cake [4] is shown in Figure – 2.5 that talks about eight

interlinked steps to ontology learning. There are three main issues involved in constructing Ontologies for a particular domain [14]:

- Associating different terms
- Building hierarchies of terms and concepts
- Identifying and labeling ontological relations

Ontology building is extracting ontological elements from the input data and building ontology from them. Manual process of building ontologies is very tedious and needs a special focus to make the whole process automated.

One of the early frameworks which talk about learning ontologies for the SW is proposed by [55]. Availability of formal ontologies with comprehensive and transportable machine understanding ensures the success of SW. Whenever an attempt is made to make ontology learning process automated, we need to answer certain questions like how fast is the ontology development process, is it really difficult and how rugged the process is in terms of accuracy and precision. Although some approaches are beneficial towards integration of machine learning techniques with knowledge acquisition process, a few other approaches have drawbacks in terms of the structure of the database.

The ontology learning framework suggested by [1] has 5-stage architecture which starts with ontology import, followed by extraction, pruning, refinement and evaluation stage.

- Stage – 1: Existing ontologies are imported and reused. This is done by mapping the structure of existing ontologies with the proposed structure of the ontology which is to be established.
- Stage – 2: Ontology extraction is carried out from the web documents.
- Stage – 3: Target ontology is pruned with a defined outline so that prime objective of the ontology building is met.
- Stage – 4: Ontology refinement is done based on the log files of user queries and generic user data.
- Stage – 5: Validation is done to check the precision and accuracy of the model.

The whole model works on iterative principle and tries to achieve better performance over a period of time. It uses N-grams statistical method for concept extraction. Association rules are extracted using modified version of generalized association rule learning algorithm which discovers the properties between the classes. A few challenges in terms of ontology boundaries and conceptual structures have been highlighted by [1].

Another novel approach for ontology learning is suggested by [63, 64]. It is one of the very early systems that use machine learning techniques and natural language processing together in a single system. It has three main phases namely terminology extraction, semantic interpretation and creating a specialized view of WordNet. Semantic interpretation is the novel aspect of this approach. Right concepts for complex domain term components are identified and semantic

relationship is extracted in order to simplify the issue of selecting the right meaning for the right concept. Decision tree C4.5 is used as algorithm for extraction of semantic relations because it provides a set of rules as output. Although the end result of the suggested framework is good, but there is a lot of scope for improvement.

These two approaches give us a head start in ontology learning. Both the frameworks are doing concept and / or feature extraction using natural language processing and / or machine learning algorithms. Association Rule Learning algorithm and Decision Tree C4.5 are being used by above techniques for rule, relation and concept extraction. Over a period of time, many researchers have proved it experimentally that SVM performs far better than these two approaches [77]. Hence there is a strong case for SVM implementation in ontology learning process.

It is important to consider association and extraction of terms and concepts by identifying and labeling ontological relations during ontology learning process [14]. Therefore, a few techniques have been suggested by researchers in the field of automatic classification with ontologies using SVM [28] which address the primary requirements.

A complete framework [52] has been proposed for automatic ontology learning that enables the process to retrieve documents from the web using focused crawling and then uses a SVM classifier to identify domain-specific documents and execute text mining in order to extract useful information for the ontology enrichment process.

In this framework, focused crawling is used to retrieve documents and information in a particular domain area. This is done as a combination of general search engines, scholarly search engines and online digital libraries. SVM is used as the most accurate classifier [31] to automatically separate out the relevant documents belonging to the domain of interest from the huge number of retrieved documents. Focused crawler is being used because it yields good recall as well as good precision, by restricting its periphery to a limited domain. A 4-step approach is suggested for automatic ontology learning.

- Step – 1: Small domain ontology is created manually by domain expert that is called seed ontology. It automatically generates queries for each and every concept available in the small but structured ontology.
- Step – 2: It utilizes the focused crawler to submit the queries generated in step – 1 to a variety of web search engines and digital libraries, and download top ranked 10 potentially relevant documents. HTML parser is being used to extract the hyperlinks.
- Step – 3: SVM classification is applied to filter-out relevant documents in the search results that match the query well and are relevant to the required ontology domain. LIBSVM classification tool is used to separate the documents into two main categories i.e. relevant and non-relevant in the context of domain.
- Step – 4: This step focuses on information extraction from the documents received in step – 3 that can be used to enrich the ontology. POS calculates weights of the tokens in relevant documents. In order to have more

accuracy, inverse document frequency (IDF) of the tokens across 10000 random documents is downloaded from the web.

Experiment is carried out on 6 different tests corresponding to the four candidate lists [52]. It is found out that step – 3 gives an average accuracy of over 77.5%. Step – 4 suggests an effective precision of 88% during information extraction experiments with SVM.

Ontologies can be quite large in certain scenarios, both in terms of concepts and the number of instances. A technique has been suggested for large-scale hierarchical text classification using SVM and coding matrices [13]. The problem of populating large ontology with a huge number of hierarchically organized classes and instances has been addressed through this approach. Classes corresponding to the concepts of ontology and instances are represented by large number of attributes. The process of ontology population is to identify the instances for a given hierarchy of concepts. Multi-class classification problem is converted into several two-class classification problems and then the results are combined to form the final classification of a document into topic hierarchy. The conversion of multi-class into two-class problem is done by coding matrix which works on the principle of 1-vs-1 or 1-vs-Rest. Two-class problem is easily solved by SVM and the collation of two-class results gives an ensemble SVM structure which helps in addressing large-scale hierarchical text classification problem.

In one of the experiments [78], three approaches (Naïve Bayes, Naïve Bayes with shrinkage and SVMs) are compared for

automatically placing reasoning facts into an ontology. SVM has outperformed the other two with good accuracy.

Extraction Mechanism	Approach
Association Rule Extraction	5 - Stage architecture focused on ontology import, extraction, pruning, refinement and evaluation.
Decision Tree C4.5	3 - Phase approach covering terminology extraction, semantic interpretation and creating a specialized view of WordNet.
SVM	Focused crawling for document retrieval and SVM for document classification.
SVM and Coding Matrices	Multi-class classification problem is converted into several two-class classification problems and then combining the results to form the final classification of a document into topic hierarchy.
Naïve Bayes, Naïve Bayes with shrinkage and SVMs	The Cyc Knowledge Base Mt Hierarchy is used on Naïve Bayes and SVMs to learn appropriate placement for assertions in Ontology.

Table – 2.3: Ontology Learning Techniques

Table – 2.3 shows different ontology learning techniques, we have considered during this survey.

There are many applications developed to incorporate ontology based document classification. We have explored a few approaches in this survey.

An approach to extract terms and concepts is suggested through a multi-level document classifier [69]. A multi-level document classifier has been implemented which is based on SVM integrated with a given domain ontology. The approach is amalgamation of two approaches wherein the document classification is sharpened by integrating powerful SVM with Ontologies based on domain knowledge to provide high level of accuracy. This approach has a great advantage of high generalization and very minimal over-fitting.

In this approach, high level of classification granularity is obtained by SVM and precision in results is obtained by semantic support. SVM provides a powerful supervised learning paradigm but it is not sufficient to classify documents which have same terms with different meaning and different terms with same meaning. The most common solution for this requirement is the integration of SVM based algorithm with the given domain Ontologies.

Information retrieval is carried out in two phases. In the first phase, SVM is implemented to carry out document classification using the domain knowledge. This helps in making ‘0’ or ‘1’ for each document wherein ‘0’ points that document does not belong to the given domain and ‘1’ points the belongingness of the document to the domain. In the

second phase, the semantic ontology is used on the documents collected in first phase and the complete set is re-classified again based upon their semantic relationships pertaining to the given domain ontology.

Although SVMs are very powerful and useful for text classification but heavy dependency on the training data is a big limitation. This limitation is complemented by the use of OBIE where ontology helps the SVM to improve the accuracy of the system. NASS uses named entities of the filtered texts to query ontology about the topic in question. Spanish Soccer League data is used for experimentation. A corpus of 1755 articles tagged with Spanish Company Grupo Heraldo is used. More than 95% recall and precision is obtained using this amalgamated model.

There is a great focus on the field of finding level of relatedness among documents available on web. This field is explored primarily because of the fact that a huge amount of documents are available on web and sometimes, search engines do not provide a desired result to a query which requires similar documents to be made available to user. A text mining technique is proposed where semantic relatedness of documents is predicted using SVM. Relatedness of documents is computed based on a specific training corpus of text documents without requiring domain-specific knowledge. The technique computes semantic relatedness and not semantic similarity. Semantic similarity is a special case of semantic relatedness.

Self-Organizing Maps (SOM) and SVMs are used for performing clustering and classification of texts. The approach follows a 2-step process.

- Step – 1 does information retrieval to encode documents with vectors where each word is mapped along with its occurrence frequency.
- Step – 2 implements SVM for text classification.

The implemented algorithm is based on multiple categorizing processes using the SVM classifiers with One-Against-All (OAA) method. It primarily judges each measure and tags it under the category of “related” or “not-related”. A threshold of relatedness is pre-defined and classification is accordingly done. This threshold value provides a degree of relatedness.

Performances of SVM-based classifiers with three kernel functions are evaluated along with ANN (Artificial Neural Network) and KNN algorithms. Three kernel functions used during the experimentation are Polynomial, Exponential and Gaussian kernels. During the experimentation, it was found that SVM performed better than rest of the algorithms. While comparing three kernel functions based SVM, it was observed that Gaussian kernel function was superior to the other two classifiers.

Table – 2.4 provides an overview of a few ontology based text classification approaches explored during this survey.

Category	Approach	Classifier	SVM Kernel Function
Multi-level Semantic Document Classifier	Amalgamation of two approaches helps in implementing multi-level semantic document classifier. SVM provides the classification supported by re-classification using semantic ontologies.	Naïve Bayes, Simple SVM, SVM with ontologies	Linear
OBIE	SVM classifier retrieves named entities to filter most relevant articles	SVM	Polynomial and RBF
Semantic Relatedness	Self-Organizing-Maps (SOMs) and SVMs are used to cluster and classify text.	ANN, KNN, SVM	Polynomial, Exponentia

	One-Against-All (OAA) method is implemented using SVM classifier to find out semantic relatedness among documents.		l, Gaussian
Text summarization	sentence extraction is done with the help of ontology based features that maps the extracted sentences to the desired nodes of a hierarchical ontology	SVM, Base classifiers	Linear

Table – 2.4: Ontology based text classification approaches

Ontology based text summarization is one of the fields where SVMs are implemented with better results vis-à-vis base classifiers. SVM is trained to identify the summary sentences with the help of ontology-based features to provide accurate results.

Main advantages of text classification using ontologies are easy extensibility of available entities, flexibility in classification process, homogeneity, and centralization of

process. Ontology learning using SVM is one of the current research areas. In the road map for web mining [4], SVM has been highlighted as one of three main classification techniques. Over a period of time, many approaches have been suggested by researchers to highlight the suitable fitment of SVMs in SW Mining related systems.

First and the foremost area in the field of SW mining is ontology learning and extraction. Automatic learning of ontology through focused crawling and information extraction provides on an average accuracy of 77.5% and 88% information extraction effectiveness which clearly shows that automatic ontology learning can be used by domain experts to enrich the existing domain ontology. But these average figures reflect that a lot of scope exists further to improve the techniques and strengthen the algorithm of the existing approach. Population of large ontologies is also one of the grey areas in the context of SVMs. A novel approach targeting large scale hierarchical ontology population opens up this field with coding matrices combined with SVM provides very encouraging results [13]. SVM performs better when compared with NB while learning ontology [78].

Next in the periphery is semantic document classification using SVMs. One basic approach for ontology mapping using SVM as a classifier shows promising results [64]. Another framework uses SVM classifier along with ontology mapping mechanism to do multi-level semantic document classification with decent accuracy. The architecture is a novel one with a lot of distinguished features that addresses multi-level classification of the documents using domain ontology. Users

can annotate web pages with Resource Description Framework (RDF) metadata effectively by identifying the semantic components conveniently using SVM. Performance of SVM and SVM integrated with domain ontology is compared where the second approach outperforms the first one. Still there is a lot of scope for improvement on precision, recall and F-measure.

Finding of level of semantic relatedness among web documents has been successfully implemented using SVM and the experiment provides very encouraging results when compared with ANN and KNN, but there is a scope of improvement in the field of finding semantic relatedness with SVMs. Similarly, text summarization is one field which requires more probing when implemented with SVM although approach provided by gives us good results.

Through this literature survey, we have tried to explore the fitment of SVMs in the field of SW Mining. Starting from the very concept of SVMs, their evolution and various kernels used in SVMs are discussed. A critical analysis is done for various semantic kernel functions used in SVMs. This covers both linear and non-linear SVM implementations. A detailed evaluation in the field of automatic ontology learning is highlighted by exploring the work done in this field.

The whole exercise reflects that SW is primarily used for high value analytical data or reasoning. In order to gain reasonable performance for semantically hosted data, we must have fine-tuned statistical learning algorithms like SVMs which along with semantic kernels can help gaining the acceptable reasoned SW. It is very much critical to enrich

ontologies not only for the semantic web but also from the SW.

In order to leverage the SW, it is most important to have a well-suited ontology set-up in place and avoid asking the SW to do more that it can do at this point. Ontology enrichment is quite reasonable expectation at this point which may be explored by applying inductive learning methods for learning probabilistic ontologies. It may help in very low error rates, ability to induce new knowledge from SW that is not logically derivable [27, 28].

Chapter – 3

KDIMS Framework – Design & Development

Main objective of this research work is to design and develop one KDIMS that is self-governed in terms of ontology population and does not require any pre-built ontology either full-fledged or seed. It all starts from user query, build a seed ontology from it and automatically enrich it by extracting concepts from the downloaded web documents only. The suggested system framework facilitates the human-machine interaction in line with the relevant search results based upon the user query and classify the downloaded documents in appropriate categories. The important point of our framework is that it does not use any pre-built ontology and starts from scratch to convert the knowledge into a powerful web document classification mechanism purely focused on user's query.

There are many frameworks which have been suggested and implemented by various researchers that describe automatic ontology learning for semantic web.

[1] suggested a comprehensive 5-step framework of ontology learning for the semantic web. The framework proceeds through importing, extracting, pruning, refining, and evaluating the ontology. They have used Text-To-Onto ontology learning environment which helps in learning from free text, from dictionaries or legacy ontologies to build and enrich a given domain ontology. The suggested framework is semi-automatic as it requires the involvement of ontology engineer to support the framework during different stages of learning.

[64] suggested OntoLearn system for automatic ontology learning from domain text. This system crawls domain specific web sites and data-warehouses for extracting terminologies, filters them using natural language processing and statistical techniques. A domain concept forest is created that provides a semantic interpretation of these terminologies duly supported by WordNet and SemCor lexical knowledge bases. The framework has three major phases namely terminology extraction, semantic interpretation and creation of WordNet specialized view. Inductive machine learning technique is being used to associate the appropriate relations among complex components of domain concept. Again, this framework is semi-automatic as it relies on WordNet and requires an involvement of ontology engineer.

[95] have suggested an ontology based approach to classify web documents. Already available knowledge base is used as a starting point. Ontology is built for each subclass of the knowledge base which uses RDFS (Resource Description Framework Schema) for transforming knowledge into ontology. A comparison between various machine learning algorithms like SVM, KNN and LSA (Latent Semantic Association) are compared with ontology based approach which clearly shows advantages of ontology based classifier. Again, this system also depends on prior available information in the form of knowledge base.

[13] suggested a framework to classify multi-class textual documents using SVM and a coding matrix. Existing ontology is populated by extracting hierarchy of concepts and instances from the large corpus of documents. Again, the main assumption here is that some “training data” is already available which

consists of primarily a set of instances with correct assignment of concepts.

[83] also provided a semi-automatic approach to enrich the vocabulary of each concept in a given ontology with words mined from the set of crawled documents and then combining with WordNet.

[52] suggested a framework for ontology learning that uses a web crawler to retrieve documents from web, identifying domain specific documents using SVM and extracting useful information from them to enrich domain specific ontology. Existing small, manually-created domain ontology is being enriched through this process by adding new concepts added in its hierarchical structure. Our approach is similar to this framework with a difference of eliminating the need of any external pre-existing domain ontology to classify the new unknown documents.

Let us now share the detailed design and development of the KDIMS framework.

3.1 Framework Design

The proposed system framework is triggered by the user input in the form of English language query sentence. This moves to query manager which in turn provides the same to two different blocks namely focused web crawler and seed ontology generator. After downloading the document corpus from the web, preprocessing is carried out which in turn becomes the input to feature extraction and selection blocks. The output of this block is again shared with two blocks in parallel: one being the ontology manager for ontology population and the other

being the SVM classifier for training purpose. After the ontology is populated, it is also provided to the training model which is replicated as testing model to finally evaluate the model performance in terms of classifying documents based on the user query. The block wise details of the system are provided in Figure – 3.1.

3.1.1 End-User Query Interface

Query from the end-user is presented to the Query Manager which is a GUI based Interface. The query interface asks for certain information from the user. It tries to collect as much information as it can in terms of concepts being shared by the user in the form of unstructured English language sentence.

3.1.2 Query Manager

This interface handles two tasks. First task is to make the query available to focused web crawler which searches relevant web pages from the web. Second task is to provide the basic information fed by the user to the block which initiates building of “Seed Ontology” from it.

3.1.3 Focused Web Crawler

It is used to retrieve documents and information from the web purely based on the query submitted by the user. The outcome of focused web crawler is a corpus of web documents which are labeled as domain specific dataset and is stored locally. This is a set of raw web documents. Although focused towards the domain of the query shared by the user, still the

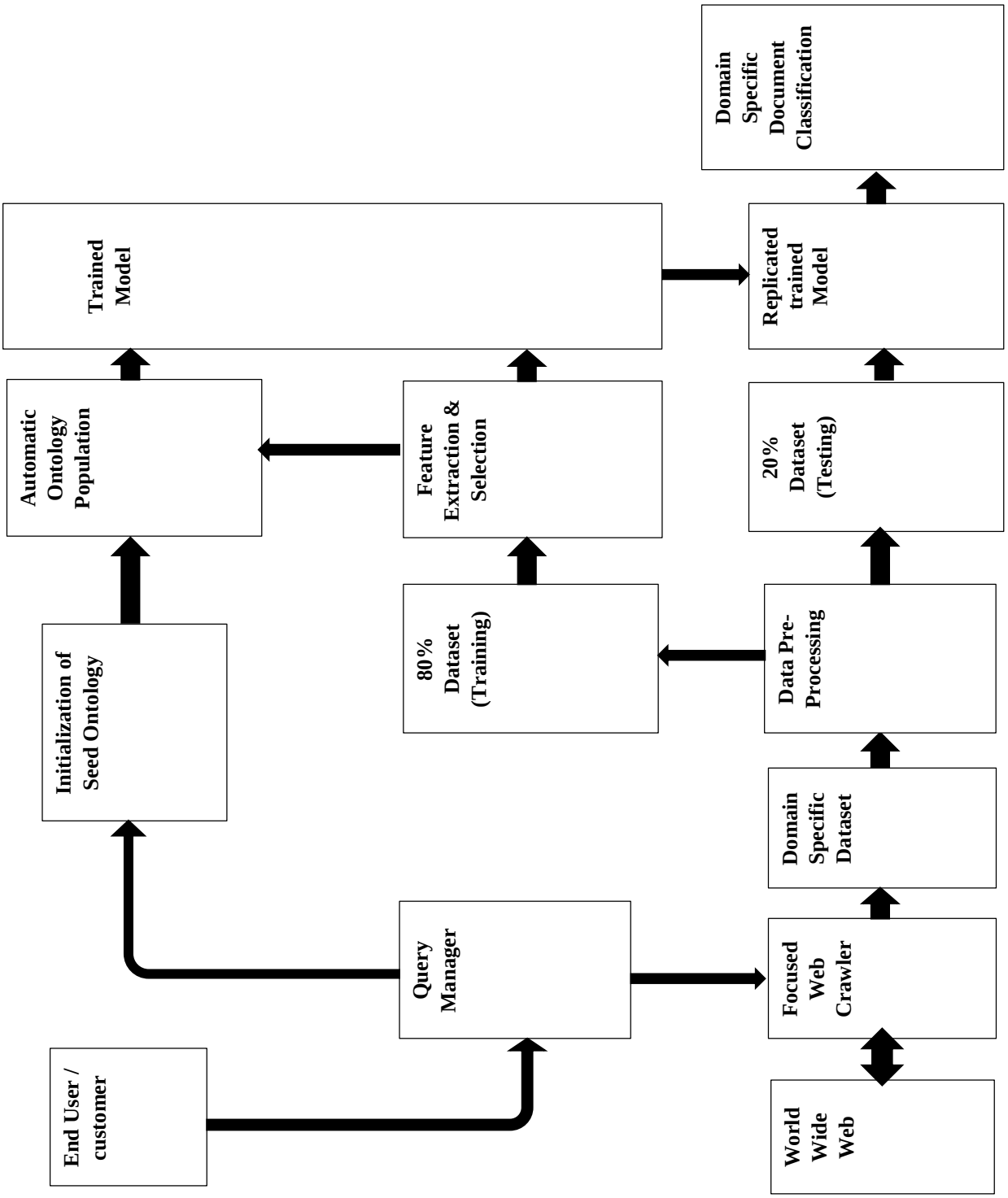


Figure – 3.1: KDIMS System Design

dataset requires preprocessing to be done before we may focus on the information extraction process.

3.1.4 Data Preprocessing

Preprocessing is done to eliminate language dependent factors. This step is very critical and mandatory prior to doing any meaningful text mining or analytics. The basic steps are:

- Scope: Choose the scope of the text to be processed. In our case, it is a set of documents.
- Tokenization: Break text into discrete words called tokens.
- Remove stop-words: Remove common words such as ‘the’, ‘they’, etc.
- Normalize spellings: Unify misspellings and other spelling variations into a single token.
- Detect sentence boundaries: mark the end of sentences.
- Normalize case: Convert the text to either all lower or all upper case.
- Stemming: remove prefixes and suffixes to normalize words – for example run, running and runs would all be stemmed to run.

Here we can define feature extraction as a combination of tokenization, stop-word removal and stemming. We have used tf-idf (term frequency – inverse document frequency) to extract features in the corpus.

Term Frequency – Inverse Document Frequency (TF-IDF) is an important technique in information retrieval which

evaluates how important is a word in a document [72]. It also plays an important role in converting the textual representation of information into a vector space model (VSM) or into sparse features. TF-IDF determines the relative frequency of the words in a specific document as compared to the inverse proportion of that word over the entire corpus of the documents under review.

Let D = Collection of documents

w = total number of terms in a document

t = a term

d = individual document where d ϵ D

We have term-frequency defined as

$$tf(t,d) = \frac{f(t,d)}{\max\{f(w,d) : w \in d\}}$$

The inverse document frequency (IDF) is defined as a measure of whether the term t common or rare across all documents.

$$idf(t,D) = log\frac{|D|}{|\{d \in D : t \in d\}|}$$

Where |D| is the total number of documents in the corpus.

$|\{d \in D : t \in d\}|$ is number of documents where the term t appears i.e. $tf(t,d) \neq 0$.

In case, the term is not in the corpus, then denominator will become “divide-by-zero”.

Therefore, we adjust the formula as $1+|\{d \in D : t \in d\}|$ to deal with this scenario.

Hence, $tf-idf$ is calculated as

$$tf-idf(t,d,D) = tf(t,d) \times idf(t,D)$$

In other words, $tf-idf$ assigns each term present in the document a weight which is

- Highest when a term t occurs many times within a small number of documents
- Lower when the term occurs fewer times in a given document, or occurs in many documents
- Lowest when the term occurs in virtually all documents

After dataset is preprocessed and ready for further processing, we split the whole set into two parts. 80% of dataset is being used for training purpose and rest 20% is being used for testing the trained model. This split is done as random and no specific technique is used.

3.1.5 Feature Selection

A minimal subset of features (extracted in the previous step) is selected so that we may realize the maximum generalization ability of the classifier. Two well established methods are available in machine learning for feature selection namely wrappers and filters [2]. Wrapper methods are very time consuming, hence have been ignored during this exercise.

Filter methods work independent of the learning algorithm that will use the selected features. During feature selection, filter method uses an evaluation metric that measures the ability of the feature to differentiate each class from the other. There are two types of filter methods namely forward selection and backward selection method. In backward selection, all the features are considered in the first instance and one feature is deleted at a time which deteriorates the selection criteria the least. We go on deleting the features till the time selection criteria reaches a particular acceptable value. In forward selection, an empty set of features is the starting point. We go on adding one feature at a time, which improves the selection criteria the most.

Selected features during this step are used for two purposes:

- One is to generate the training model with SVM as the learning algorithm.
- Second one is to serve these selected features as inputs to the ontology manager which populates the ontology in the context of user query.

3.1.6 Seed Ontology Generator

This interface takes the preprocessed user query from Query Manager as input. This input is converted into a basic ontology tree using Resource Description Framework (RDF) graph [69]. Any given RDF graph contains a collection of triples; each consists of a combination of Subject – Predicate – Object (S – P – O). Each triple is extracted from a given sentence

which reflects the relationship between its subject and object linked by a predicate.

A sample RDF graph is designed as below:

```
< ?xml version = "1.0"? >

< rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >

< rdf:Description about = "http://www.whitehouse.gov/~BarackObama/" >

.

.

< /rdf:Description >

< rdf:Description rdf:ID = "Barack Obama" >

.

.

< /rdf:Description >

< /rdf:RDF >
```

Sample RDF graph for User Query String “Barack Obama”

We term it as seed ontology equipped with a very small number of classes. This is the starting point of enriching the ontology for the specific search query written by the user.

3.1.7 Ontology Manager

This is the most critical block of our whole system. The success of the system depends on how this block populates the ontologies in the form of RDF triples from the given set of documents downloaded from the web using focused web crawler. The process flow for Ontology Manager is as shown in the Figure – 3.2.

The process starts with preprocessed text documents as input. Named Entities are recognized and relations are extracted to primarily mark subjects, objects and predicates followed by RDF translation of S-P-O. This block takes two inputs: one being the seed ontology prepared by seed ontology generator and features extracted by Feature extraction block.

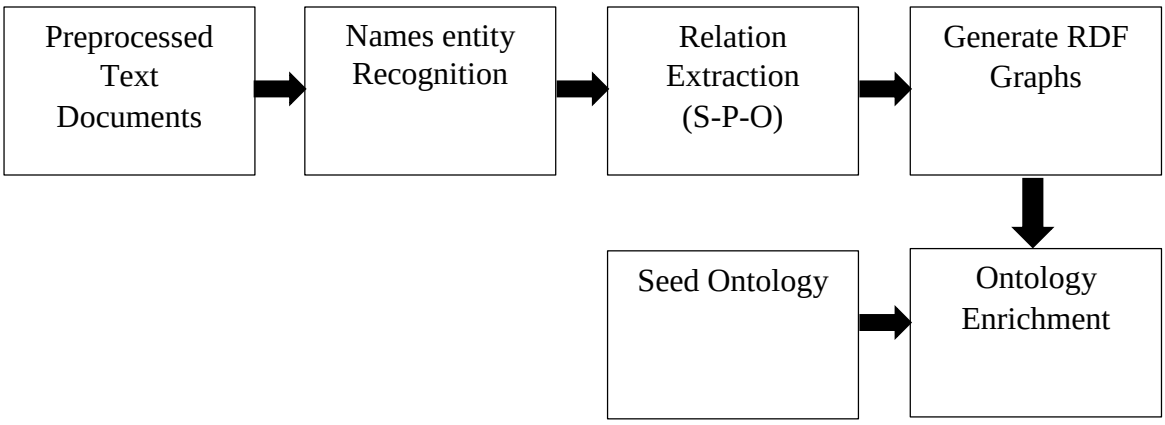


Figure – 3.2: Ontology Manager Framework

3.1.8 Training and Testing Model

Training model is built with two flavors: one being built using machine learning algorithms and the second one being built using the ontology prepared by ontology manager. Then this model is replicated as testing model to check the accuracy and performance of the model using un-known dataset. Document classification is being done according to the user based query and categorized in appropriate groups.

3.2 KDIMS Implementation

The system is being implemented using LIBSVM [16]. We have done experiments with below mentioned two setups:

- SVM based classifier with linear kernel
- SVM based classifier with self-populated ontology

10 - Fold Cross Validation (CV) is used during the experimentation stage. 10 - Fold CV primarily breaks the given data into 10 sets of equal sized subsets, train on 9 datasets and test on 1 dataset. The whole process is repeated 10 times and the accuracy of the classification process is calculated by taking the mean of all the stages. 10 - Fold CV is a useful way for accuracy estimation and model selection [44, 67].

The proposed system is evaluated using two use-cases, first being evaluated on three offline datasets and other being evaluated using online dynamic dataset downloaded from the web.

3.2.1 Use-Case1: Offline classification

Three benchmark datasets namely Reuters-21578 [50] (Dataset available at http://www.daviddlewis.com/resources/testcollections/reuters21578/) , 20-Newsgroups collection [49] (Dataset available at http://qwone.com/~jason/20Newsgroups/) WebKB [19] (Dataset available at http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/) are used for offline classification.

- Reuters-21578 is currently the most widely used test collection for text categorization research. We have 6532 documents as training dataset and 2568 documents as test dataset.
- The 20 Newsgroups data set is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. The data is organized into 20 different newsgroups, each corresponding to a different topic. We have segregated the whole dataset in two parts with 11293 documents as training dataset and 7528 documents as test dataset.
- WebKB is a collection of web documents collected by the World Wide Knowledge Base (Web -> Kb) project of the CMU text learning group, and were downloaded from the 4 Universities Data Set Homepage. These pages were collected from computer science departments of various universities in 1997, manually classified into four main classes: student, faculty, course, and project. The dataset is divided into two parts with 2803 documents forming the training dataset and 2396 documents forming the test dataset.

3.2.2 Use-Case2: Online classification

We have done the experiment with 5 different user query strings. Focused web crawler downloads top 40 pages each from five search engines namely Google, Bing, Yahoo, AltaVista, and AOL. 160 pages were preprocessed and then used for feature extraction and ontology population. Remaining 40 pages were thrown to the testing model for run time classification. Table – 2 provides the top 5 categories of documents for each of the user query string.

User Query String	Top Category - 1	Top Category - 2	Top Category - 3	Top Category - 4	Top Category – 5
Barack Obama	Wikipedia	Personal Web Site	White House	Biography	News
US President Barack Obama	White House	Wikipedia	Personal Web Site	News	Biography
Barack Obama holiday in Hawaii	News	Wikipedia	--	--	--
Barack Obama visited China	News	Wikipedia	--	--	--
Sandy	Sandy Hurricane - News	Sandy Hook School - News	Sandy - City	Sandy - Wikipedia	Sandy - social networking sites

Table – 3.1: User Query response under Use-Case2

It is very much clear that “Wikipedia” is the top classified set of documents when the query string says “Barack Obama”. It changes to “White House” when the query string is modified as “US President Barack Obama”. Query 3 and 4 also points that when user is trying to search some happening, top category of classified documents changes to “News”. Fifth query gives some uneven results because of the query string meaning. It is clear from the top 5 categories that although we got related “News” under top 2 categories, the classification system provides an option to the user to choose the classification folder accordingly to his / her choice. This particular output – 5 is encouraging for us where we have provided the novice user an opportunity to choose from these five categories when “Sandy” is searched.

3.3 Ontology Manager Implementation

This is achieved by writing 5 RDF graphs for five user queries as mentioned in Table – 3.1.

```
< ?xml version = "1.0"? >

< rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >

< rdf:Description about = "http://www.whitehouse.gov/~BarackObama/" >

.

.

< /rdf:Description >
```

```
< rdf:Description rdf:ID = "Barack Obama" >

.

.

< /rdf:Description >

< /rdf:RDF >
```

Code Snippet – 1: RDF graph for User Query String "Barack Obama"

```
< ?xml version = "1.0"? >

< rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >

< rdf:Description about = "http://www.whitehouse.gov/~BarackObama/" >

.

.

< /rdf:Description >

< rdf:Description rdf:ID = "US President Barack Obama" >

.

.

< /rdf:Description >

< /rdf:RDF >
```

Code Snippet – 2: RDF graph for User Query String "US President Barack Obama"

```
< ?xml version = "1.0"? >

< rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >

< rdf:Description about = " URL fetched from Web crawler " >

.

.

< /rdf:Description >

< rdf:Description rdf:ID = "Barack Obama holiday in Hawaii" >

.

.

< /rdf:Description >

< /rdf:RDF >
```

Code Snippet – 3: RDF graph for User Query String "Barack Obama holiday in Hawaii"

```
< ?xml version = "1.0"? >

< rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >

< rdf:Description about = " URL fetched from Web crawler " >

.

.
```

```
< /rdf:Description >

< rdf:Description rdf:ID = "Barack Obama visited China" >

.

.

< /rdf:Description >

< /rdf:RDF >
```

Code Snippet – 4: RDF graph for User Query String "Barack Obama visited China"

```
< ?xml version = "1.0"? >

< rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >

< rdf:Description about = "URL fetched from Web crawler" >

.

.

< /rdf:Description >

< rdf:Description rdf:ID = "Sandy" >

.

.

< /rdf:Description >

< /rdf:RDF >
```

Code Snippet – 5: RDF graph for User Query String "Sandy"

These RDF graphs formulate the core of ontology manager. Seed ontology is developed using the user query fed through query manager. Pre-processed text documents are fed to the relation extraction mechanism that helps in providing S-P-O triples as input to ontology enrichment block. This way, we enrich the ontology purely based on the user query, starting from scratch. This is one of the highlighted feature of our framework.

This also gives us a flexibility to enrich ontology irrespective of its domain as it is domain independent. Therefore, this framework may find itself quite useful in those business scenarios where users are novice and would like to build the knowledge base from scratch.

Chapter – 4

Performance & Accuracy Analysis of the proposed system

Accuracy is the degree of conformity of a measured quantity to its true value, while precision is the degree to which further measurements show similar results. In other words, the precision of an experiment is a measure of the reliability of the experiment whereas the accuracy of an experiment is a measure of how closely the experimental results agree with a true value.

We have done a detailed performance and accuracy analysis of the proposed KDMIS framework in this chapter. We have implemented the system with two use-cases, one with offline datasets and one with dynamic web dataset based on user queries. Various parameters are evaluated based on the given performance metrics. In our case, we have compared both accuracy as well as precision parameters to provide a holistic insight into the experimental outcomes under different use-cases.

4.1 Performance Metrics:

The following performance measures are evaluated:

P = the number of relevant documents classified as relevant (True Positive),

Q = the number of relevant documents classified as not relevant (True Negative),

R = the number of not relevant documents classified as relevant (False Negative),

S = the number of not relevant documents classified as not relevant (False Positive).

Therefore, the total number of documents $T = (P + Q + R + S)$

The performance measure parameters are primarily precision, recall, F-measure.

Precision = Number of correctly identified items as percentage of number of items identified

$$Precision = P / (P + S)$$

Recall = Number of correctly identified items as percentage of the total number of correct items

$$Recall = P / (P + R)$$

F-measure = Weighted average of precision and recall

$$F\text{-}measure = (2 * Precision * Recall) / (Precision + Recall)$$

Accuracy = degree of conformity of a measured quantity to its true value

$$Accuracy = (P + Q) / T$$

MCC (Matthews Correlation Coefficient) is a correlation coefficient between the observed and predictive binary

classification [57], returning a value between +1 and – 1 with + 1 provides perfect prediction and – 1 provides total disagreement [20].

$$MCC = (P * Q - S * R) / sqrt ((P + S) (P + Q) (S + R) (Q + R))$$

True Positive Rate (TPR) = Number of true positives divided by the total number of positives

$$TPR = P / (P + S)$$

False Positive Rate (FPR) = Number of false positives divided by the total number of negatives

$$FPR = S / (Q + R)$$

AUC (Area Under the receiver operator Curve) depicts a trade-off between benefits of the system (i.e. True Positives) and cost overhead to the system (i.e. False Positives). RoC (Receiver Operator Curve) is drawn with TPR on Y-axis and FPR on x-axis.

4.2 Use-case 1 experimentation

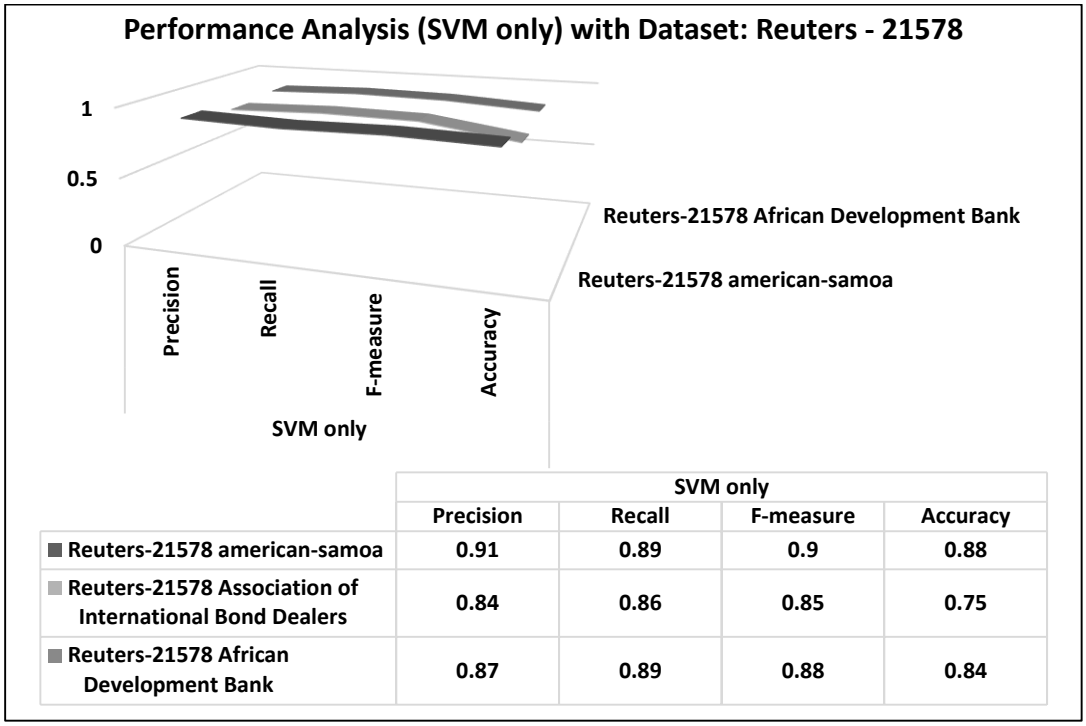
Below results have been achieved with three offline datasets and four major parameters like Precision, Recall, F-measure and Accuracy have been calculated for two set-ups mainly SVM only and SVM with system built ontology.

Dataset	User Query String	SVM only				SVM + System built Ontology			
		Precision	Recall	F-measure	Accuracy	Precision	Recall	F-measure	Accuracy
Reuters-21578	american-samoa	0.91	0.89	0.9	0.88	0.9	0.88	0.89	0.92
	Association of International Bond Dealers	0.84	0.86	0.85	0.75	0.86	0.87	0.86	0.89
	African Development Bank	0.87	0.89	0.88	0.84	0.86	0.85	0.85	0.78
20-Newsgroups	Lexan Polish	0.88	0.9	0.89	0.81	0.86	0.85	0.85	0.88
	NASA	0.92	0.91	0.91	0.86	0.9	0.91	0.9	0.92
	What is a squid?	0.85	0.84	0.84	0.74	0.87	0.89	0.88	0.81
WebKB	Course at Cornell	0.81	0.79	0.8	0.72	0.85	0.84	0.84	0.74

	Raymond J. Mooney	0.76	0.78	0.77	0.65	0.8	0.79	0.79	0.91
	Internet Softbot	0.77	0.75	0.76	0.66	0.78	0.79	0.78	0.88

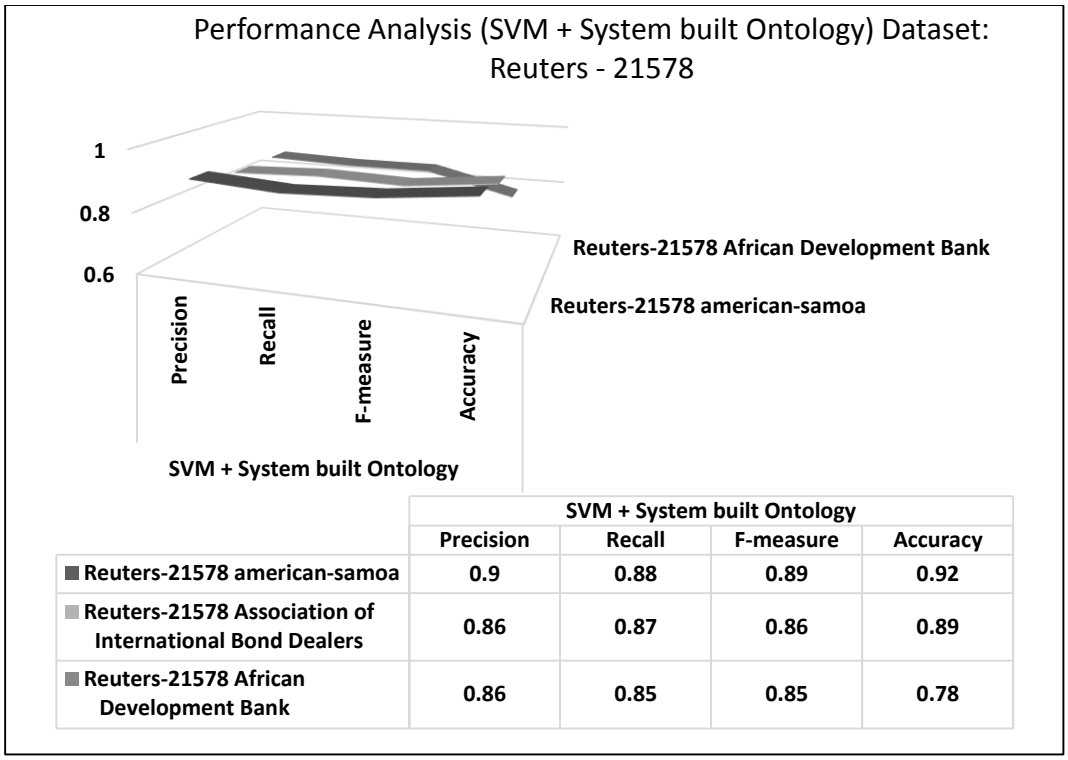
Table – 4.1: Performance metrics and Accuracy of 3 Benchmark DBs under User-Case1

Below graphs (Figure – 4.1 through 4.6) reflects the comparative values of precision, recall, F-measure and accuracy for three datasets.



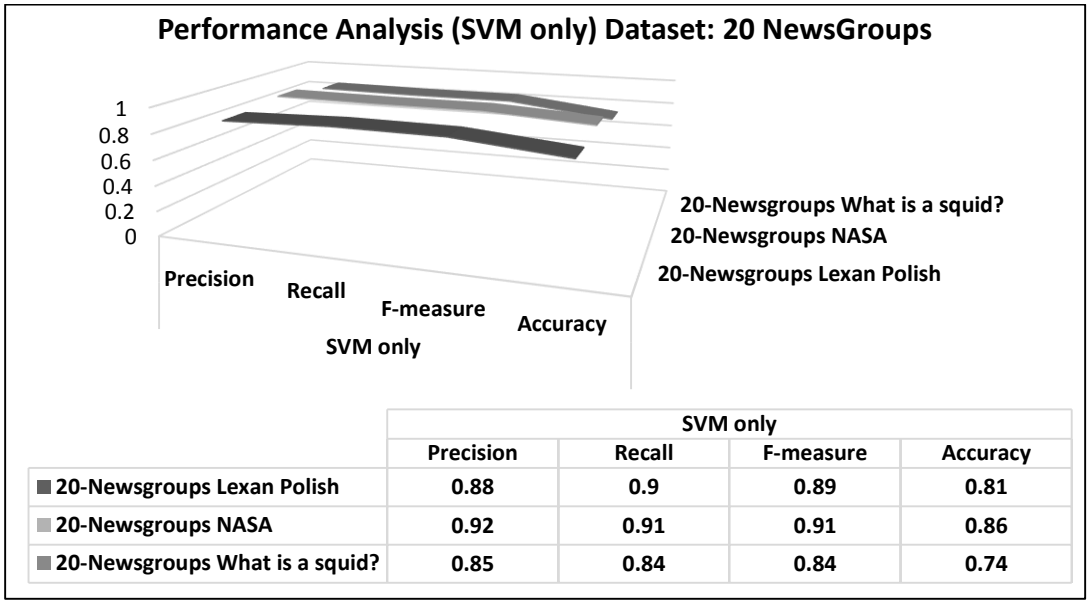
	SVM only			
	Precision	Recall	F-measure	Accuracy
Reuters-21578 american-samoa	0.91	0.89	0.9	0.88
Reuters-21578 Association of International Bond Dealers	0.84	0.86	0.85	0.75
Reuters-21578 African Development Bank	0.87	0.89	0.88	0.84

Figure – 4.1: Performance Analysis (SVM Only) with Reuters-21578 Dataset



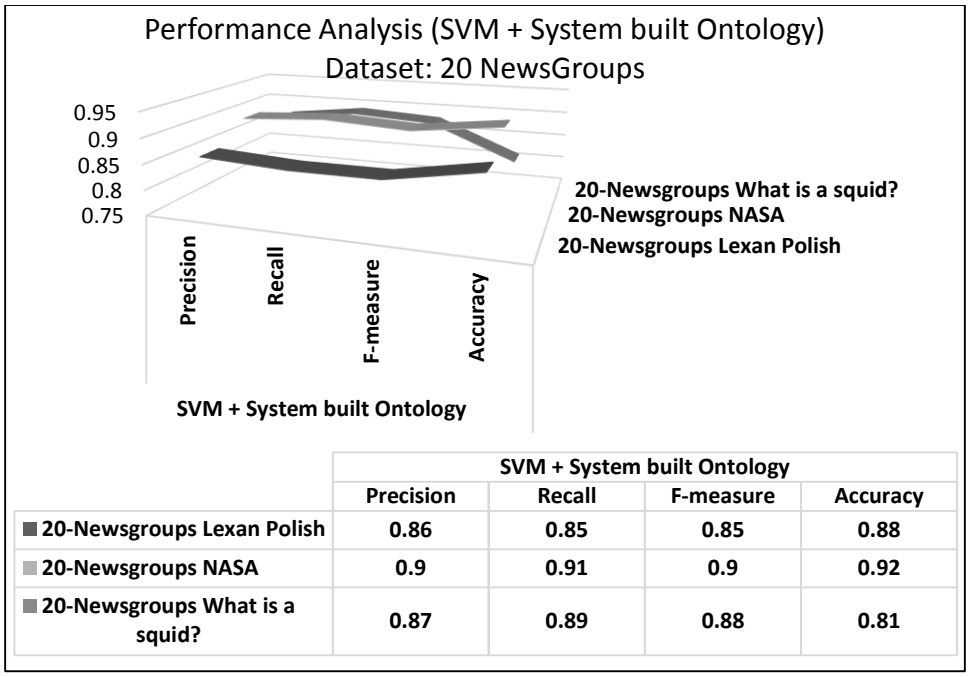
	SVM + System built Ontology			
	Precision	Recall	F-measure	Accuracy
Reuters-21578 american-samoa	0.9	0.88	0.89	0.92
Reuters-21578 Association of International Bond Dealers	0.86	0.87	0.86	0.89
Reuters-21578 African Development Bank	0.86	0.85	0.85	0.78

Figure – 4.2: Performance Analysis (SVM + System built Ontology) with Reuters-21578 Dataset



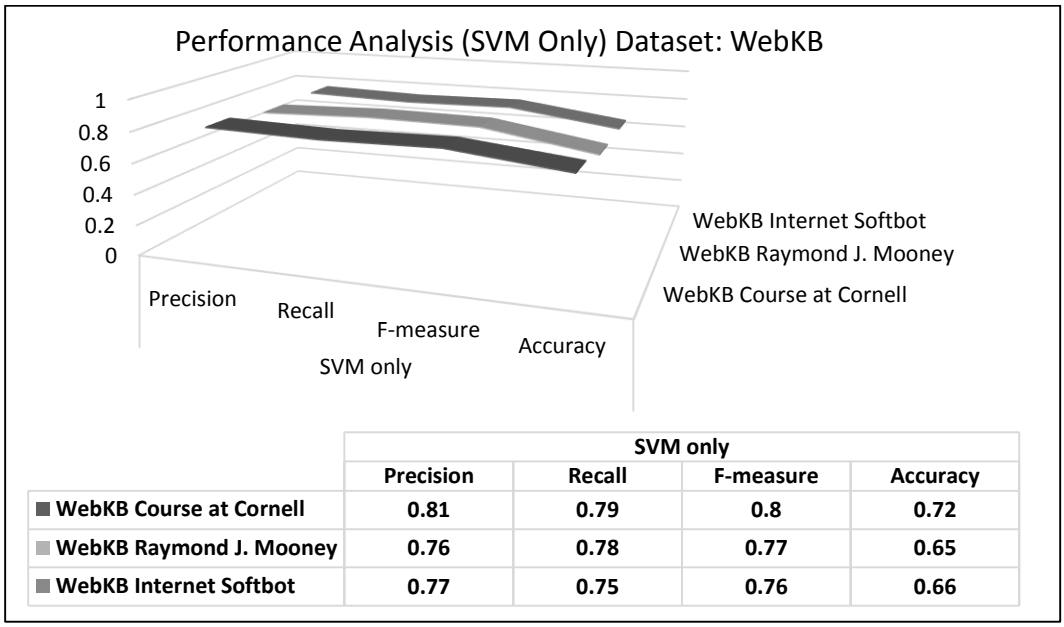
	SVM only			
	Precision	Recall	F-measure	Accuracy
20-Newsgroups Lexan Polish	0.88	0.9	0.89	0.81
20-Newsgroups NASA	0.92	0.91	0.91	0.86
20-Newsgroups What is a squid?	0.85	0.84	0.84	0.74

Figure – 4.3: Performance Analysis (SVM Only) with 20 NewsGroups Dataset



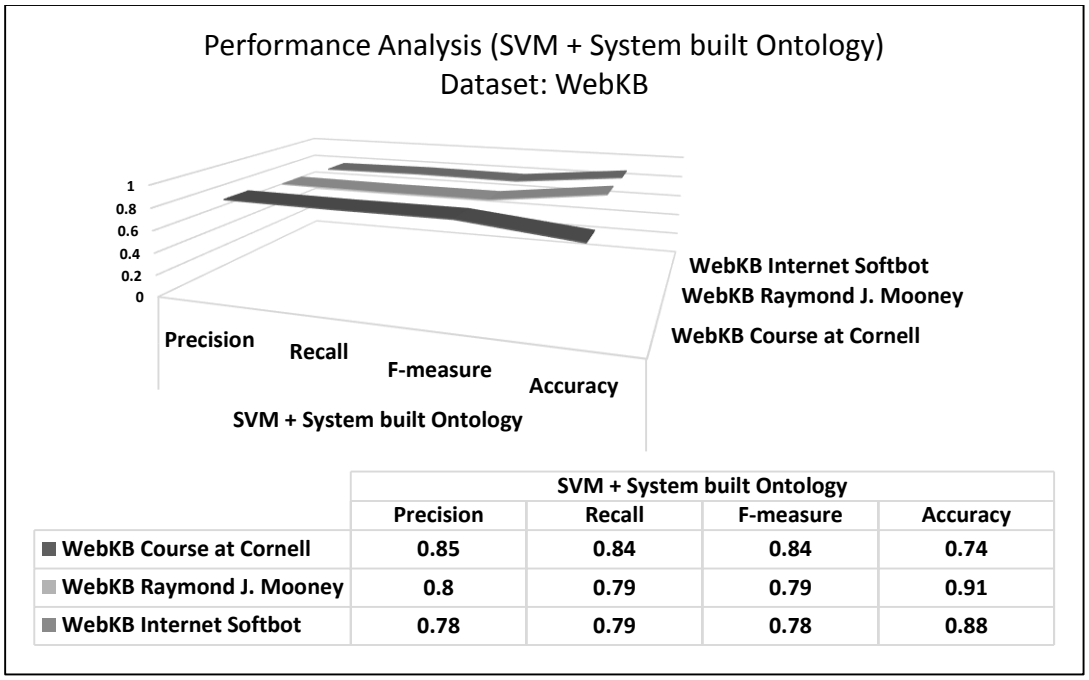
	SVM + System built Ontology			
	Precision	Recall	F-measure	Accuracy
20-Newsgroups Lexan Polish	0.86	0.85	0.85	0.88
20-Newsgroups NASA	0.9	0.91	0.9	0.92
20-Newsgroups What is a squid?	0.87	0.89	0.88	0.81

Figure – 4.4: Performance Analysis (SVM + System built Ontology) with 20-NewsGroups Dataset



	SVM only			
	Precision	Recall	F-measure	Accuracy
WebKB Course at Cornell	0.81	0.79	0.8	0.72
WebKB Raymond J. Mooney	0.76	0.78	0.77	0.65
WebKB Internet Softbot	0.77	0.75	0.76	0.66

Figure – 4.5: Performance Analysis (SVM Only) with WebKB Dataset



	SVM + System built Ontology			
	Precision	Recall	F-measure	Accuracy
WebKB Course at Cornell	0.85	0.84	0.84	0.74
WebKB Raymond J. Mooney	0.8	0.79	0.79	0.91
WebKB Internet Softbot	0.78	0.79	0.78	0.88

Figure – 4.6: Performance Analysis (SVM + System built Ontology) with WebKB Dataset

It is clear from Table – 4.1 and subsequent Figures – 4.1 through 4.6 that SVM with 10 – fold CV appears to be better than the ontology based classifier in some scenarios where the user query is straight and simple. As we go on adding the complexity to the user query, our proposed model built on ontology performs comparatively well. Average F-measure (85% for SVM with system built ontology, 84% for SVM only) and accuracy (86% for SVM with system built ontology, 77% for SVM only) parameters are much better in case of SVM with system built ontology framework as compared to simple SVM based system.

4.3 Use-case 2 experimentation

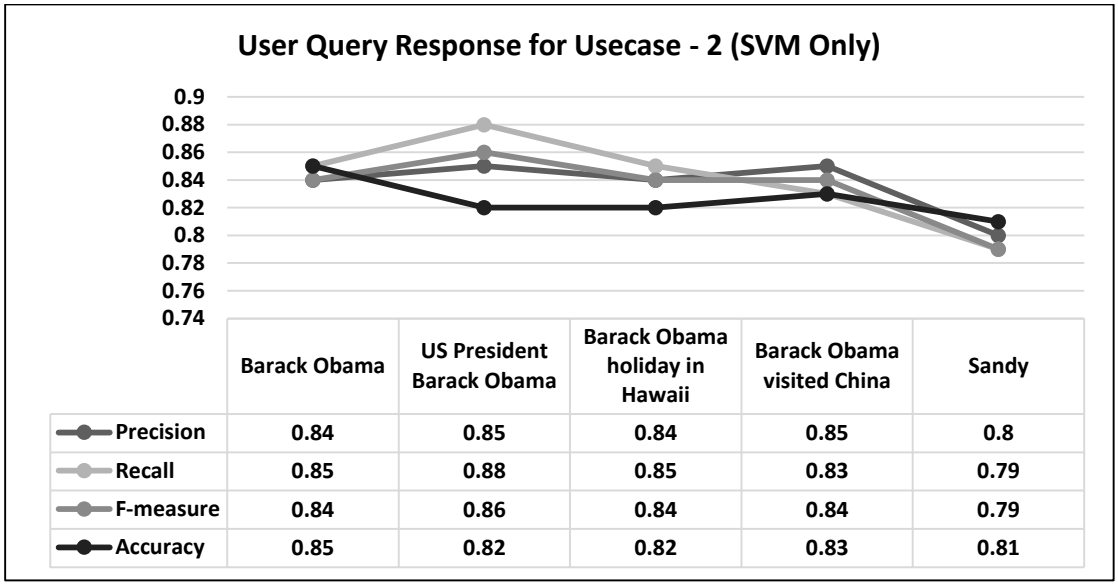
Below results have been achieved with dynamic data coming from the web and four major parameters like Precision, Recall,

F-measure and Accuracy have been calculated for two set-ups mainly SVM only and SVM with system built ontology.

	SVM only				**SVM + System built Ontology**			
User Query String	Precision	Recall	F-measure	Accuracy	Precision	Recall	F-measure	Accuracy
Barack Obama	0.84	0.85	0.84	0.85	0.83	0.84	0.83	0.89
US President Barack Obama	0.85	0.88	0.86	0.82	0.85	0.83	0.84	0.91
Barack Obama holiday in Hawaii	0.84	0.85	0.84	0.82	0.85	0.86	0.85	0.82
Barack Obama visited China	0.85	0.83	0.84	0.83	0.86	0.84	0.85	0.83
Sandy	0.8	0.79	0.79	0.81	0.82	0.81	0.81	0.82

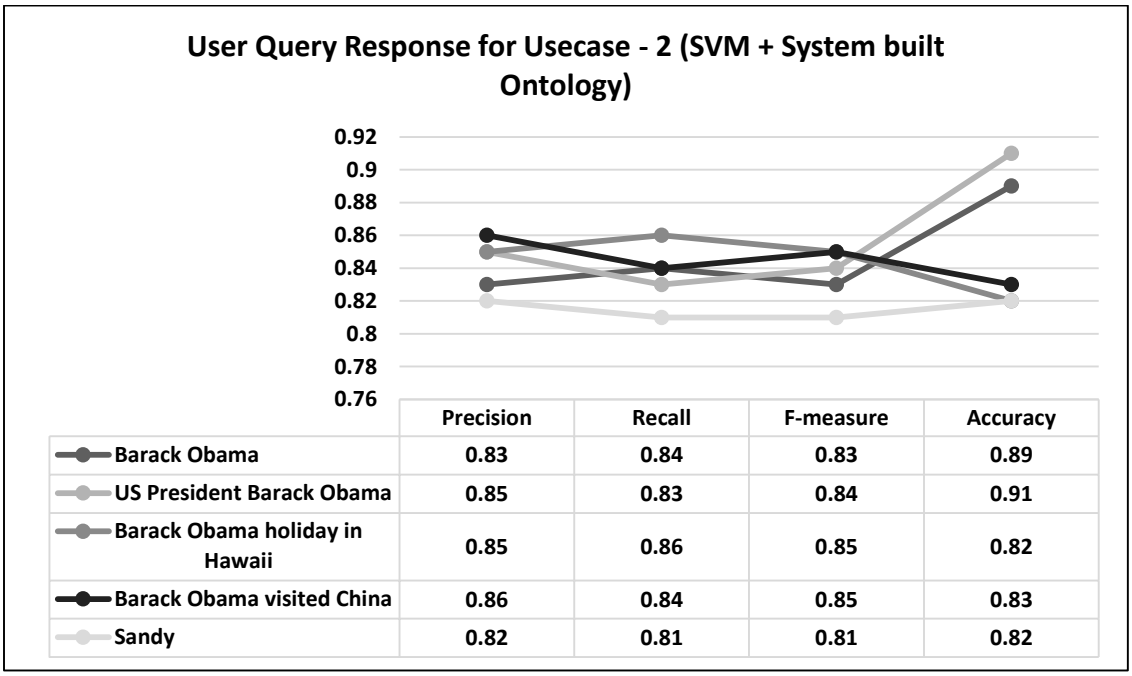
Table – 4.2: Performance metrics under use-case2

Performance analysis is carried out with the help of line graphs as shown in Figure – 4.7 and Figure – 4.8 for SVM only and SVM with System built Ontology respectively.



	Barack Obama	US President Barack Obama	Barack Obama holiday in Hawaii	Barack Obama visited China	Sandy
Precision	0.84	0.85	0.84	0.85	0.8
Recall	0.85	0.88	0.85	0.83	0.79
F-measure	0.84	0.86	0.84	0.84	0.79
Accuracy	0.85	0.82	0.82	0.83	0.81

Figure – 4.7: System (SVM Only) Response to user query



	Precision	Recall	F-measure	Accuracy
Barack Obama	0.83	0.84	0.83	0.89
US President Barack Obama	0.85	0.83	0.84	0.91
Barack Obama holiday in Hawaii	0.85	0.86	0.85	0.82
Barack Obama visited China	0.86	0.84	0.85	0.83
Sandy	0.82	0.81	0.81	0.82

Figure – 4.8: System (SVM + System built Ontology) Response to user query

4.4 Analysis

Results of ontology based approach for both user-cases i.e. with offline datasets and live datasets from web reflect that the proposed system performs reasonably well while classifying documents. The comparison of average performance and accuracy parameters for both the classifiers (SVM and SVM with system built ontology) is shown in Figure – 4.9. Average MCC values calculated from various performance parameters are as shown in Table- 4.3. All positive values of MCC (> 0) depict a reasonable representation of quality predictions as received from the experimental setup.

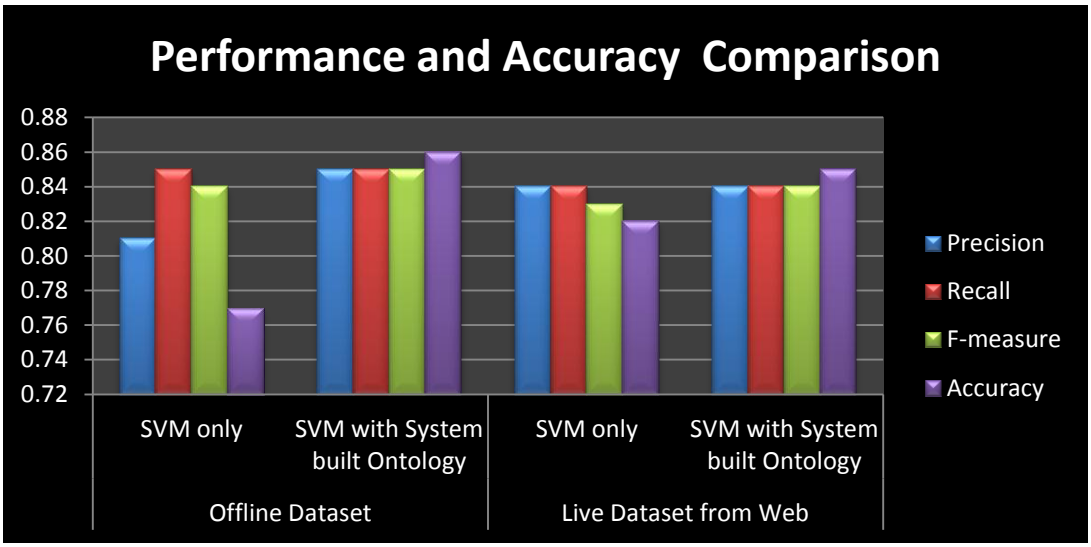


Figure – 4.9: Comparison of performance and accuracy of two classifiers

Average MCC Value		
	SVM Classifier	**SVM Classifier with System built Ontology**
Usecase1	0.33	0.62
Usecase2	0.63	0.69

Table – 4.3: MCC metrics

High accuracy of the system is also evident from the convex RoC under AUC analysis of two classifiers as shown in Figure – 4.10 (usecase1) and Figure – 4.11 (usecase2). RoC curves of both use-cases highlights that SVM Classifier with system built ontology provides more accuracy vis-à-vis simple SVM based classifier. RoC for SVM only under usercase1 is quite uneven while the RoC for SVM with system built ontology provides a smooth convex curve which is very promising. Both RoCs for usecase2 are smooth but the curve for SVM with system built ontology is more convex than the other.

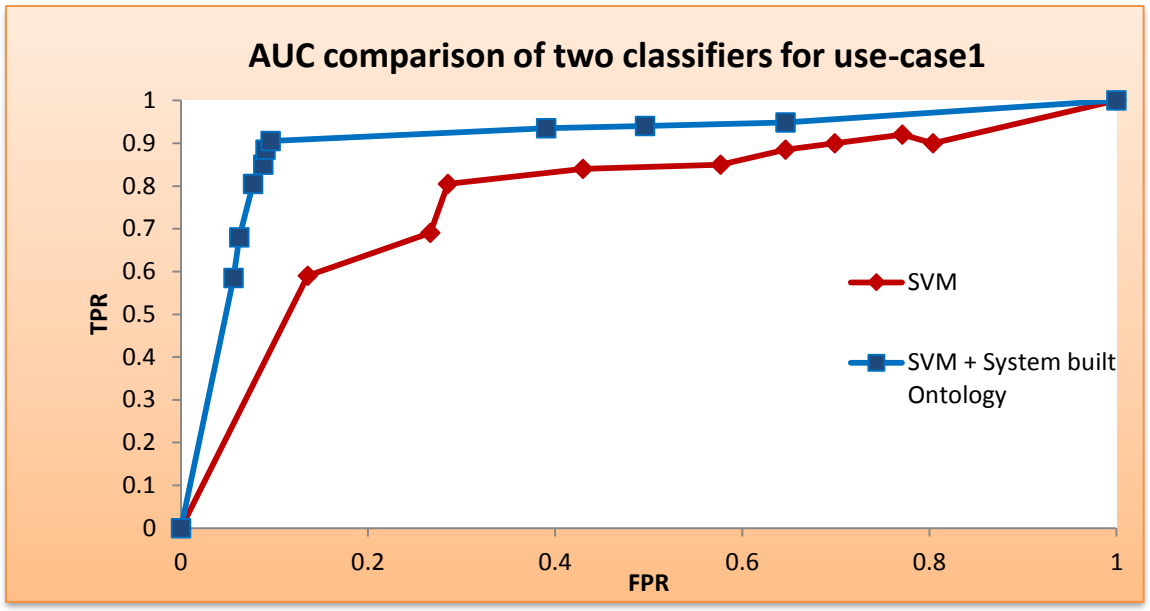


Figure – 4.10: AUC Comparison of two classifiers for use-case1

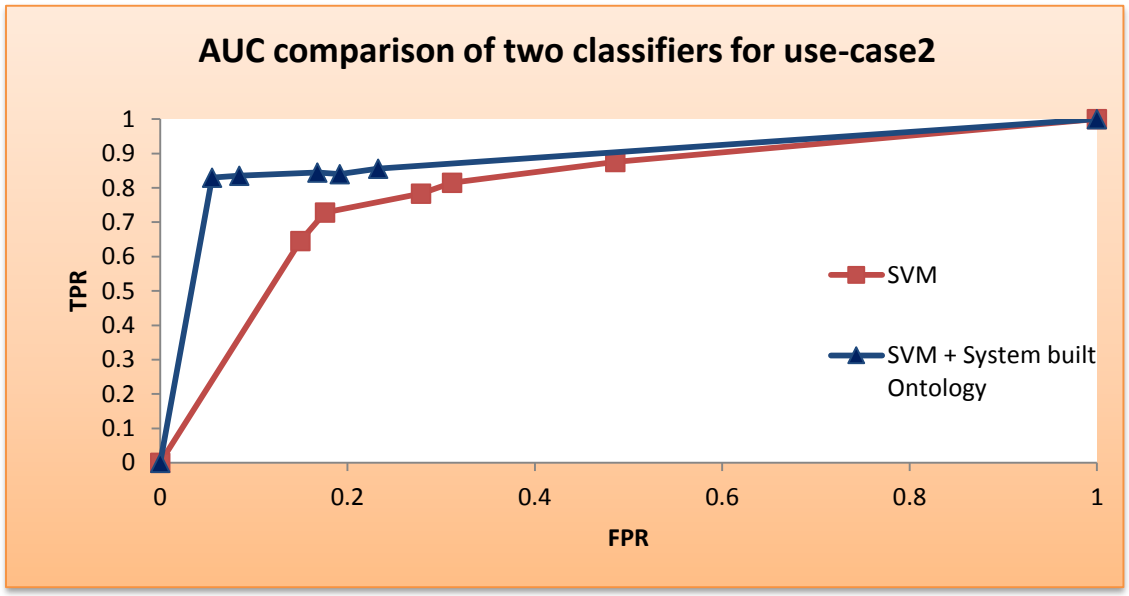


Figure – 4.11: AUC Comparison of two classifiers for use-case2

Overall, the proposed self-governed ontology based classification system provides us very favorable results particularly in scenarios where the user feeds a natural language query to the system instead of single word query.

4.5 Comparison with other methodologies

Our framework is similar to the techniques such as suggested by [52] in the manner ontology is learned and populated. Both the frameworks are similar in terms of many points like using ‘focused crawling’ for retrieving documents from web, automated ontology learning process with SVM as classification method. But there are a few critical differences also in both the frameworks.

- The first and the foremost difference is the use of ‘seed ontology’. While [52] uses a seed ontology, our framework starts from a scratch and builds the seed ontology from the user queries. This highlights a step further in building a self-governed ontology based classification system.
- The second and more critical difference between these two frameworks is the manner queries are generated and processed. While [52] implements a total automated query generation from the seed ontology to populate / enrich it further, our framework works in accordance with user query, builds seed ontology and further enrich it automatically. So our framework is more towards serving real world users in the forefront handling their queries at run time and providing them with relevant results as compared to [52] which serves to background enrichment of ontologies which will help users in future while finding more appropriate results for their queries.

- The third difference between the two frameworks is their focus area of learning. While [52] is focused towards biological domain area, our framework is independent of any specific domain area and purely focused towards user query picking up the domain at run time. Pre-fixing of domain also requires [52] to generate seed ontology first and then proceed towards enrichment process. Our framework does not have such dependencies which results in making it as 'self-governed' learning system.

The experimental analysis of both the systems also highlights quite a few contrasts. While [52] have used a threshold of '0.6' during the experimental set-up, our framework uses a threshold of '1'. This implies that [52] have considered top 60% of results while we have considered complete 100% results during performance and accuracy parameter calculations. If we use some threshold value, it will definitely help us improve our precision and accuracy more than what we have received under current set-up. The overall F-measure and precision parameters of [52] (Precision 88%, F-Measure 85%) are slightly better than ours (Precision 85%, F-measure 84%). This clearly shows a scope of improvement for our framework which we shall try to achieve by incorporating a suitable threshold parameter in our experimental set-up.

Chapter – 5

System Validation with Semantic Kernel Functions

Previous chapters show the complete implementation of KDMIS with two use cases – one being implemented with SVM only and other being implemented with SVM supported by system built ontologies. In both the cases, we have used linear kernel function for SVM. In this chapter, we shall validate the experimentation results achieved earlier by implementing the same system with Semantic kernel functions. We implement the system with four popular semantic kernel functions instead of linear kernel function. This validation exercise and experimentation provides us a unique opportunity to test our KDMIS using semantic kernels instead of syntactical linear kernel functions.

5.1 Semantic Kernel Functions

Four semantic kernel functions (Semantic Smoothing Kernel, Latent Semantic Kernel, Semantic WordNet-based Kernel and Semantic Smoothing Kernel having Implicit Superconcept Expansions) being implemented with SVM as kernel method.

The algorithms of all these four semantic kernel functions are explained in literature survey i.e. chapter - 2.

All the four semantic kernels are implemented in SVMlight [46]. 10 - Fold Cross Validation (CV) is used during the

experimentation stage. 10 - Fold CV primarily breaks the given data into 10 sets of equal sized subsets, train on 9 data sets and test on 1 data set. The whole process is repeated 10 times and the accuracy of the classification process is calculated by taking the mean of all the stages. 10 - Fold CV is a useful way for accuracy estimation and model selection [47]. Under the umbrella of Performance Metrics, following performance measures are evaluated:

5.1.1 Experimentation with Movie Review Dataset

The first experiment is done using the Movie Review data set version v2.0 [66]. This collection contains 2,000 reviews of movies from the Internet Movie Database archive. Half of the reviews express a positive sentiment about the movie, and half express a negative sentiment. Various performance parameters are compared as shown in Table- 5.1.

Data set	Semantic kernel	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	RoC Area	Accuracy
Movie review	Semantic Smoothing Kernel	0.745	0.494	0.601	0.745	0.665	0.258	0.608	0.667
	Latent Semantic Kernel	0.827	0.191	0.812	0.827	0.82	0.636	0.894	0.819
	Semantic WordNet Based Kernel	0.808	0.192	0.809	0.808	0.808	0.617	0.897	0.808
	Kernel with Implicit Superconcept Expansions	0.841	0.16	0.841	0.841	0.84	0.681	0.841	0.841

Table – 5.1: Performance parameters of semantic kernel functions with Movie review data set

Below graphs as shown in Figure 5.1 to 5.4 provides a comparison of various kernel functions with Movie Review Dataset.

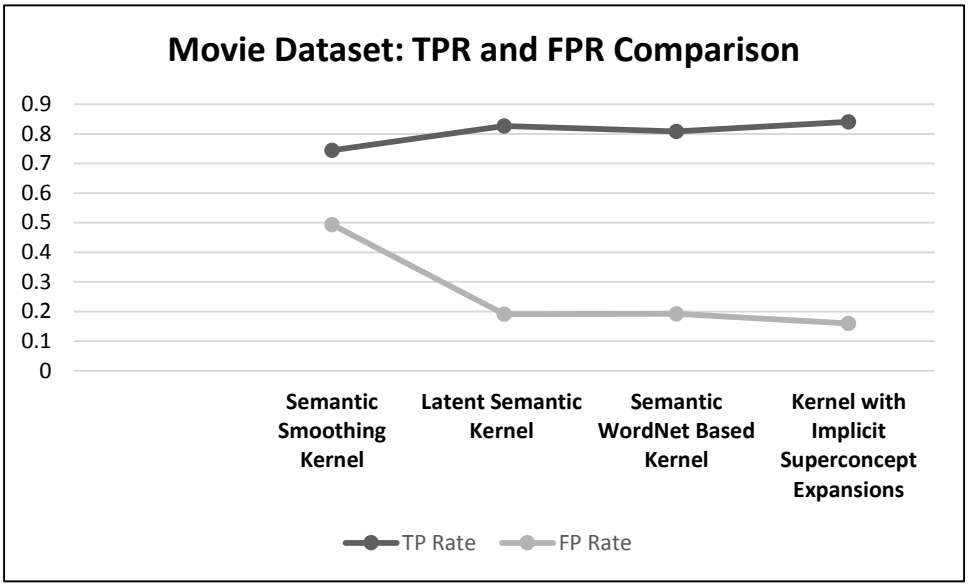


Figure – 5.1: TPR & FPR Comparison – Movie Dataset

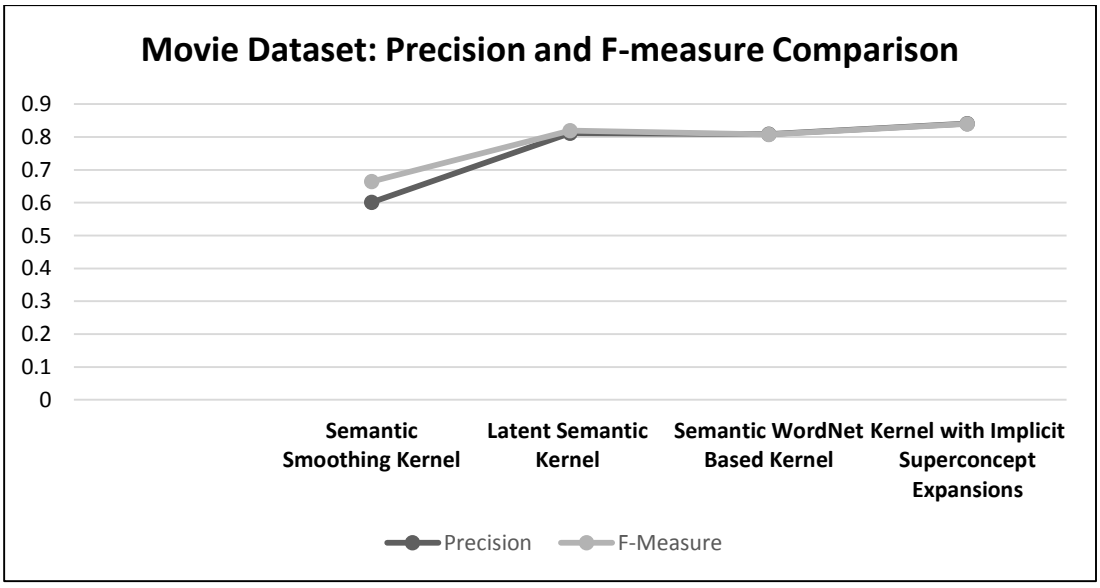


Figure – 5.2: Precision & F-Measure Comparison – Movie Dataset

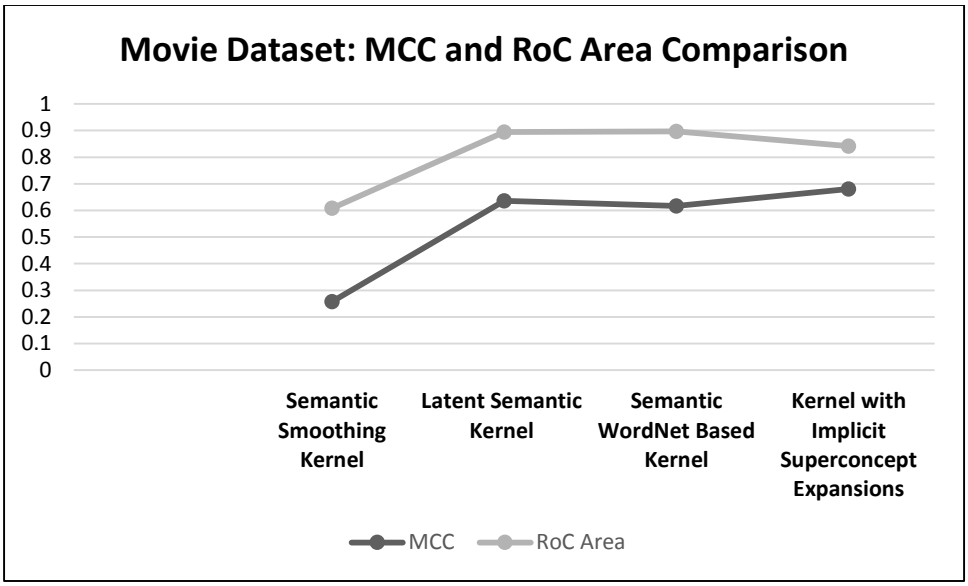


Figure – 5.3: MCC & RoC Comparison – Movie Dataset

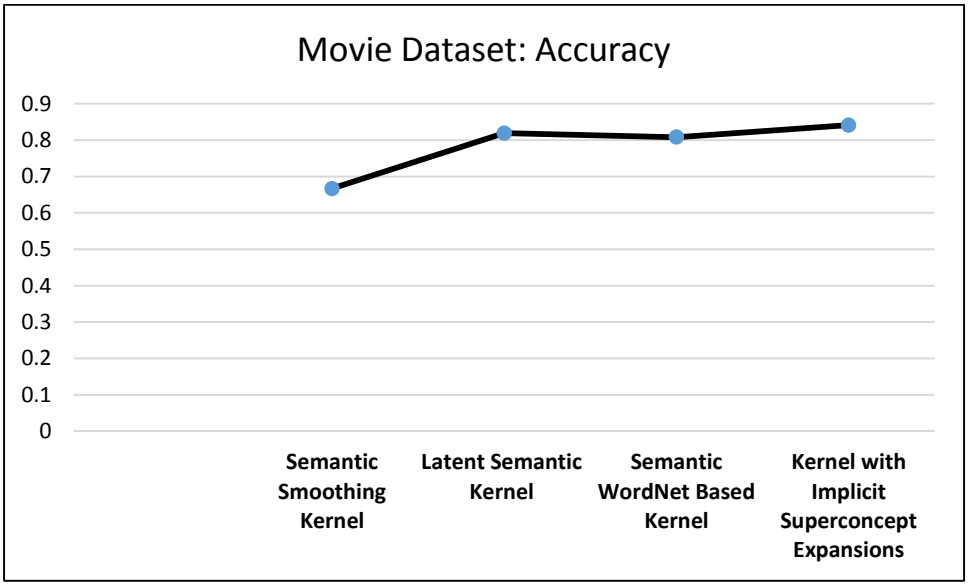


Figure – 5.4: Accuracy Comparison – Movie Dataset

Figures – 5.1 through 5.4 reflects that kernel with implicit superconcept expansions scores over the other three kernels on performance and accuracy parameters. But RoC parameter for

this kernel function is less as compared to LSK and Semantic WordNet based Kernel.

5.1.2 Experimentation with Reuters-21578 Dataset

The second set of experimentation is done using Reuters-21578 data set [71]. This collection of documents, appeared on the Reuters newswire in 1987, is one of the most widely used for text categorization research. The documents were assembled and indexed with categories by Reuters Ltd. and Carnegie Group, Inc. In our experimental set-up, we have taken 1000 positive documents and 1000 negative documents. Table - 5.2 shows the experimental points obtained from Reuters-21578 data set.

Data set	Semantic kernel	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	RoC Area	Accuracy
Reuters-21578	Semantic Smoothing Kernel	0.766	0.482	0.614	0.766	0.681	0.293	0.703	0.682
	Latent Semantic Kernel	0.81	0.22	0.786	0.81	0.798	0.59	0.795	0.798
	Semantic WordNet Based Kernel	0.834	0.166	0.834	0.834	0.834	0.668	0.911	0.834
	Kernel with Implicit Superconcept Expansions	0.897	0.103	0.9	0.897	0.897	0.797	0.955	0.896

Table – 5.2: Performance parameters of semantic kernel functions with Reuters-21578 data set

Figure 5.5 to 5.8 provides comparison of various performance and accuracy parameters for Reuters – 21578 Dataset.

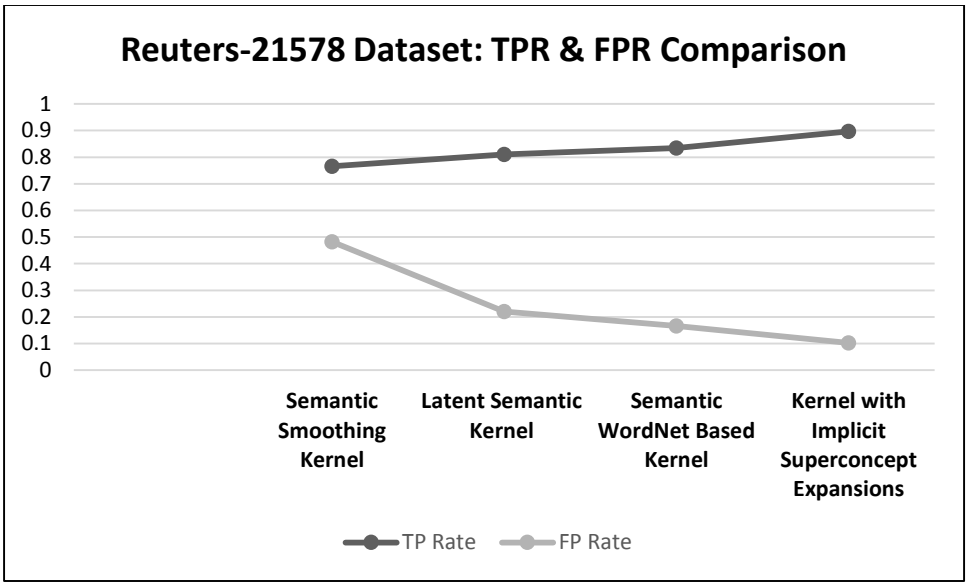


Figure – 5.5: TPR & FPR Comparison – Reuter-21578 Dataset

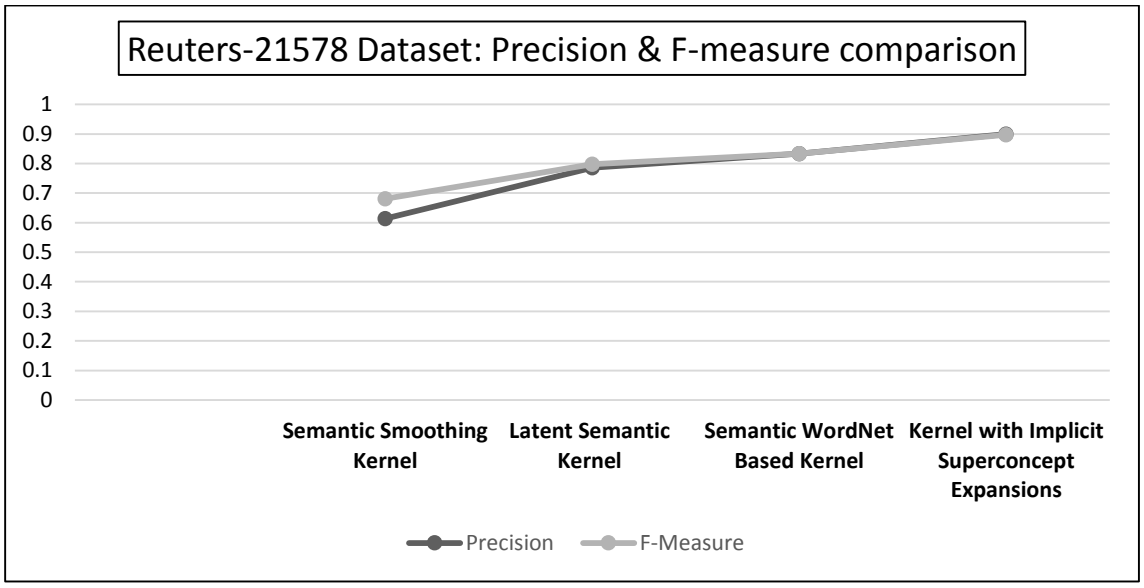


Figure – 5.6: Precision & F-Measure Comparison – Movie Dataset

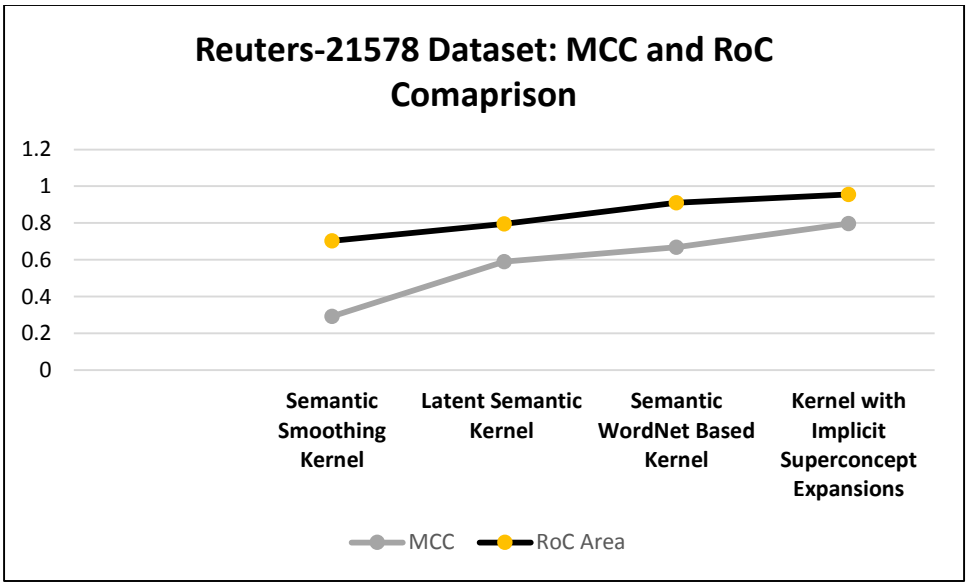


Figure – 5.7: MCC & RoC Comparison – Reuters-21578 Dataset

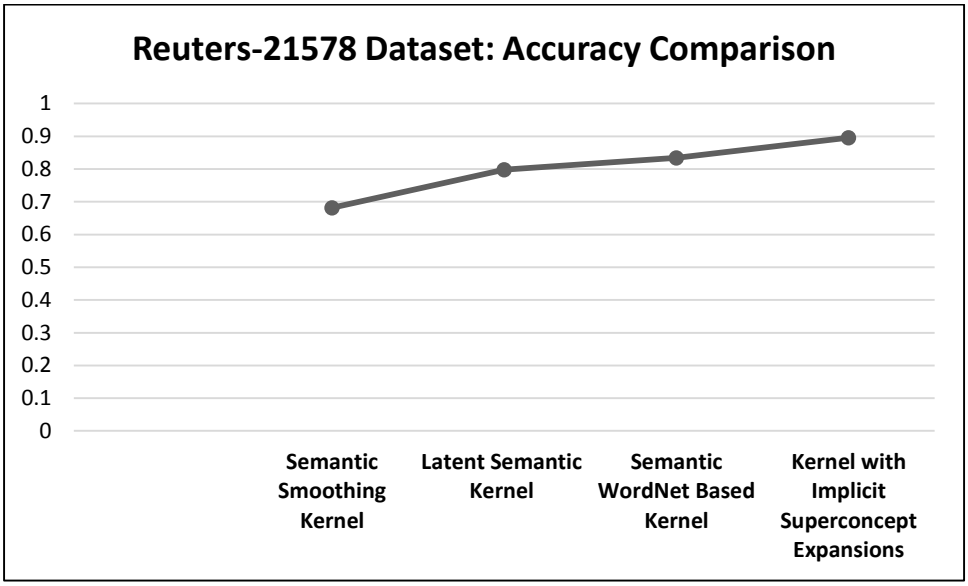


Figure – 5.8: Accuracy Comparison – Reuters-21578 Dataset

When we analyze the performance and accuracy parameters for Reuters-21578 dataset, kernel with implicit superconcept expansions is better than others in all parameters.

5.1.3 Experimentation with 20-NewsGroup Dataset

The third experiment is done using the 20-News-groups data set [49]. The 20 news-groups data set is a popular collection of approximately 20,000 newsgroup documents, partitioned nearly evenly across 20 different news-groups (about 1,000 documents per class). For classification purpose, we have chosen 1000 documents from a given news category as positive and randomly selected 1000 documents from other categories as negative. Various experimental data points are shown in Table – 5.3.

Data set	Semantic kernel	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	RoC Area	Accuracy
20 news-groups	Semantic Smoothing Kernel	0.779	0.264	0.746	0.779	0.762	0.515	0.842	0.763
	Latent Semantic Kernel	0.838	0.17	0.831	0.838	0.835	0.668	0.911	0.835
	Semantic WordNet Based Kernel	0.819	0.182	0.819	0.819	0.818	0.637	0.894	0.819
	Kernel with Implicit Superconcept Expansions	0.853	0.059	0.935	0.853	0.892	0.797	0.955	0.892

Table – 5.3: Performance parameters of semantic kernel functions with 20 news-groups data set

Graphs shown in Figure – 5.9 till 5.12 shows various performance and accuracy parameters for 20 news-groups dataset.

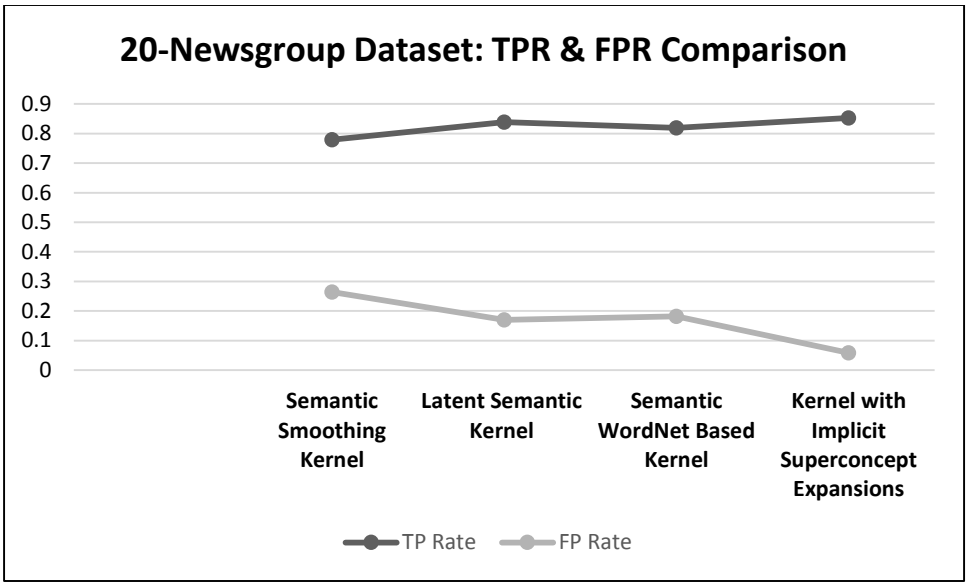


Figure – 5.9: TPR & FPR Comparison – 20-News-groups Dataset

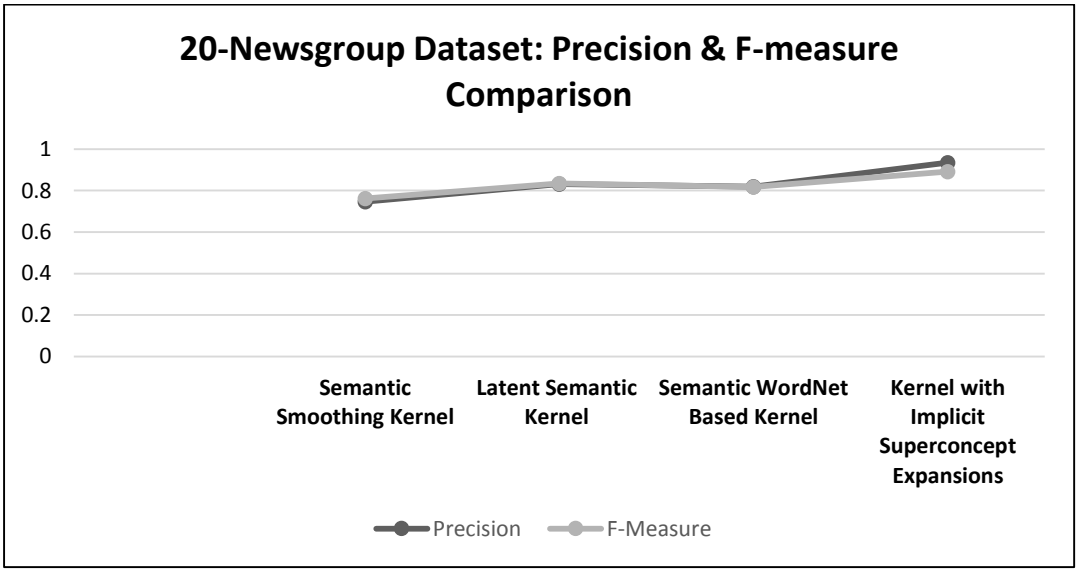


Figure – 5.10: Precision & F-Measure Comparison – 20-News-groups Dataset

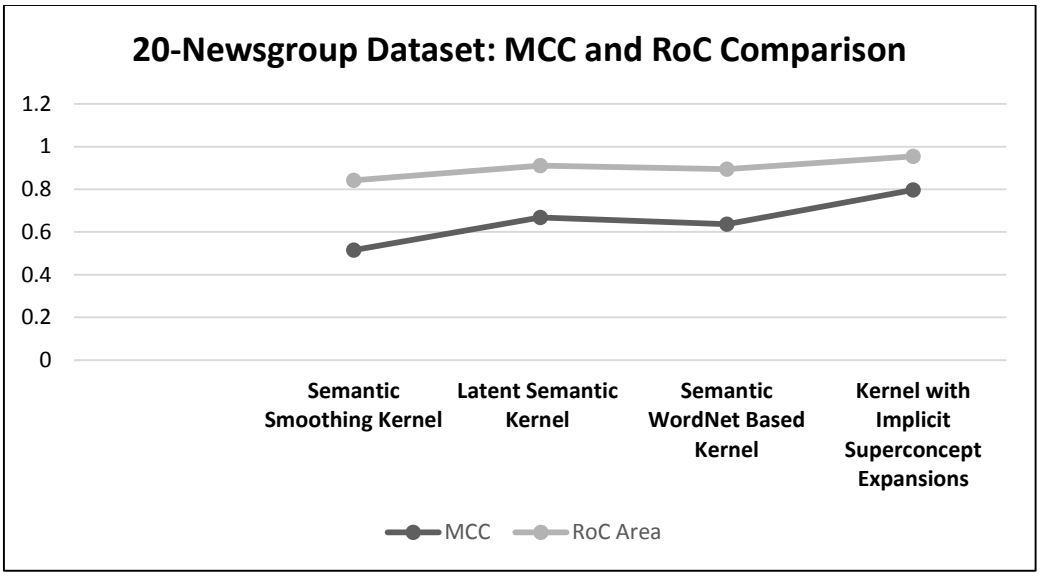


Figure – 5.11: MCC & RoC Comparison – 20-News-groups Dataset

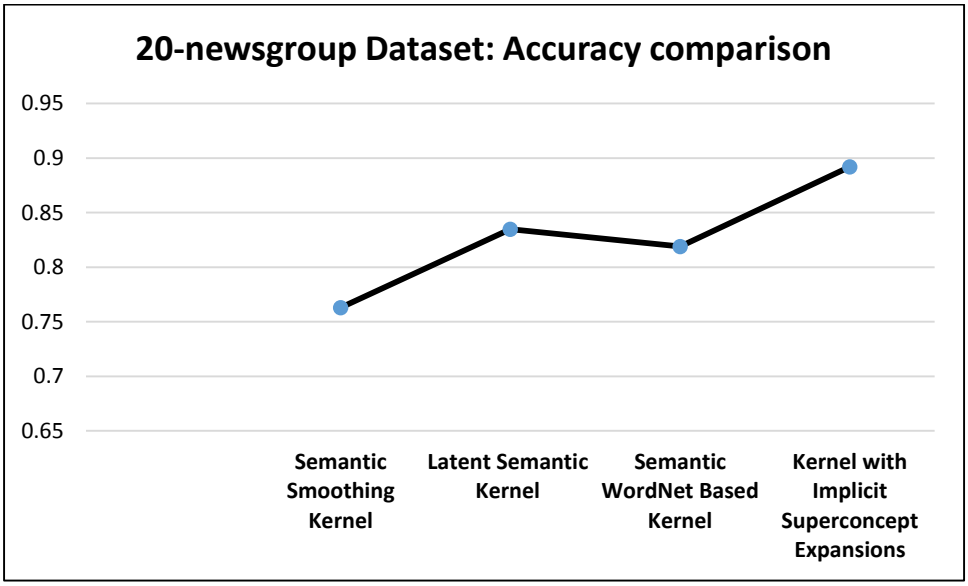


Figure – 5.12: Accuracy Comparison – 20-News-groups Dataset

Here, also the same kernel function scores over the other ones.

5.2 Result Analysis

Experimental results highlight the comparison between various semantic kernels as shown in Figure 1-4.

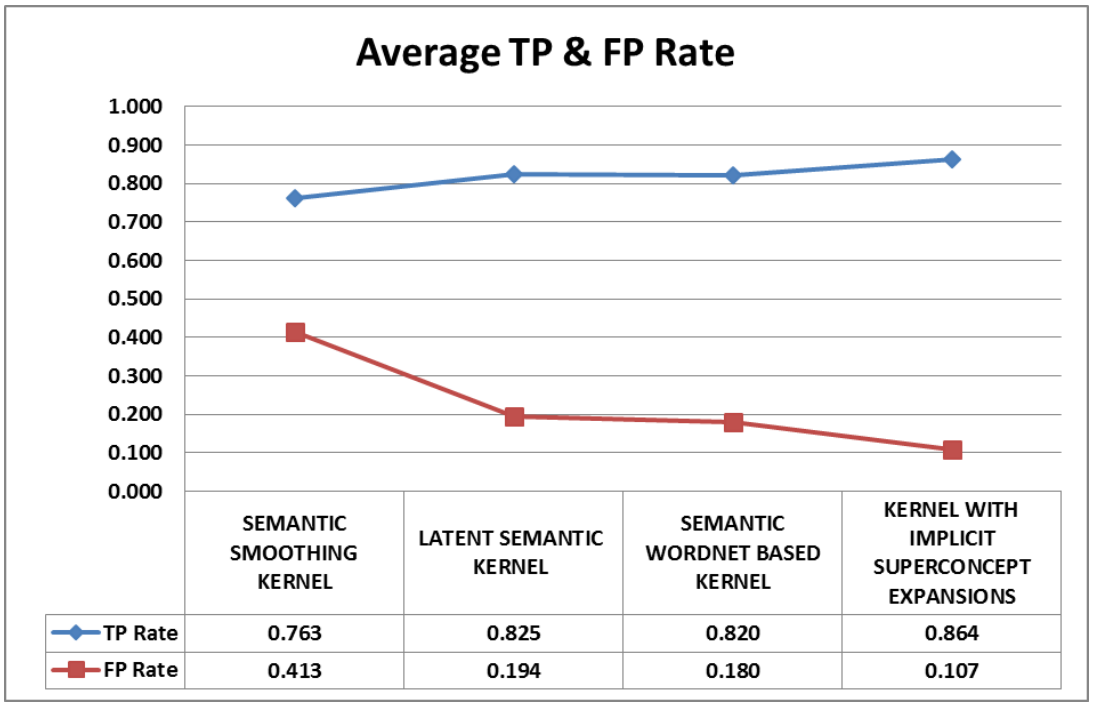
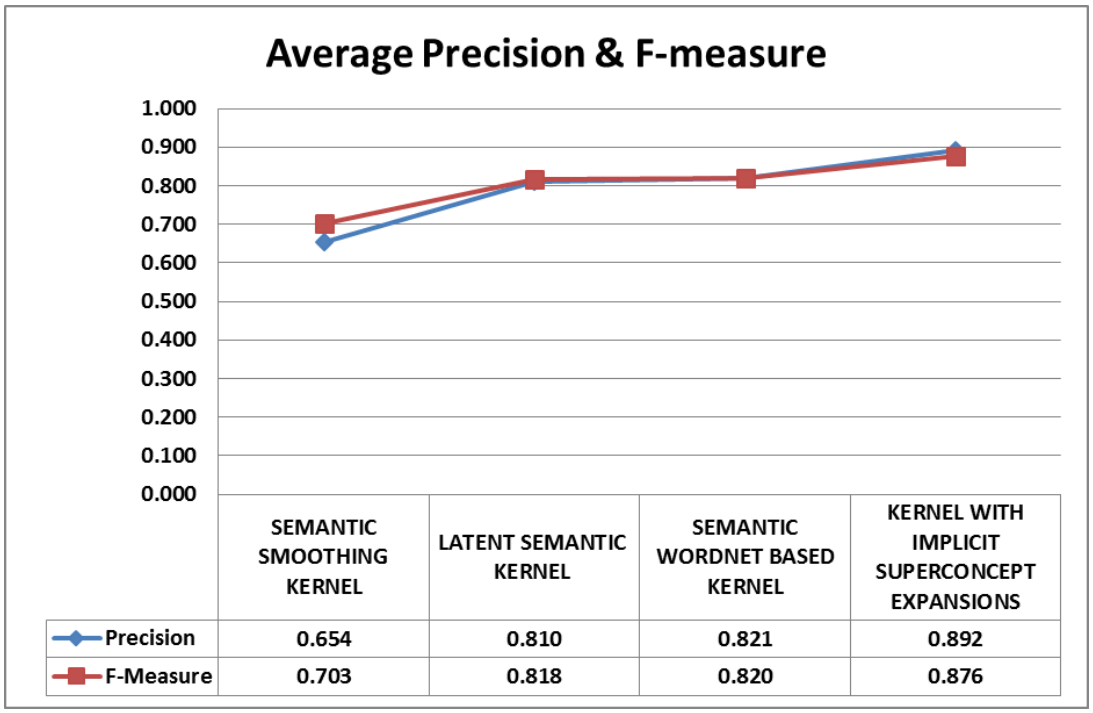
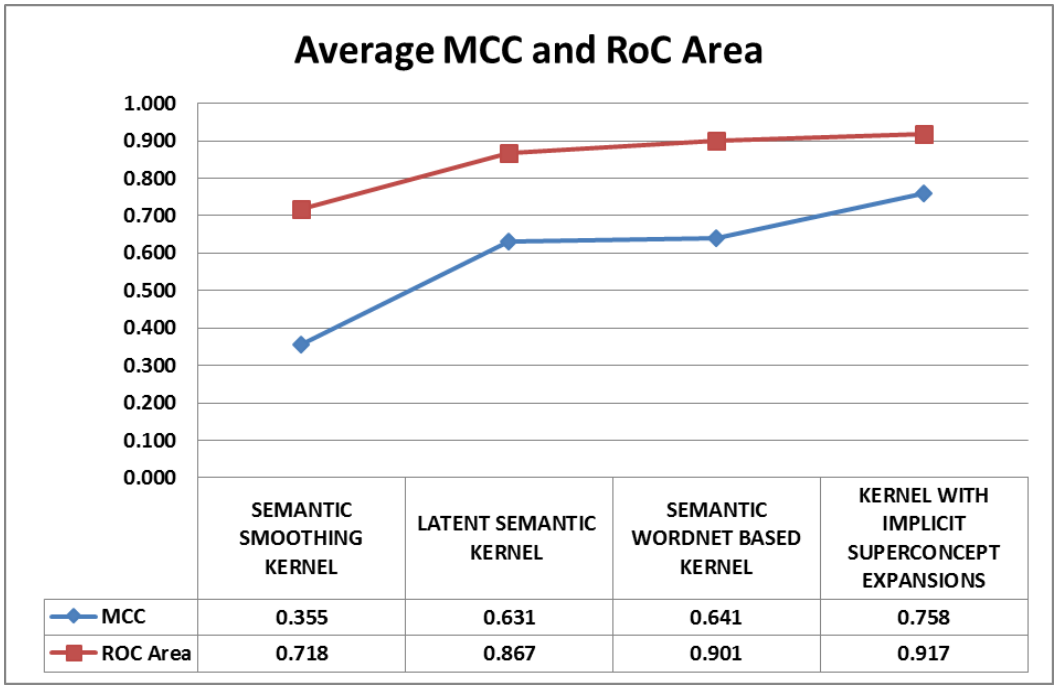


Figure – 5.13: Comparison of TPR and FPR of four semantic kernel functions



	SEMANTIC SMOOTHING KERNEL	LATENT SEMANTIC KERNEL	SEMANTIC WORDNET BASED KERNEL	KERNEL WITH IMPLICIT SUPERCONCEPT EXPANSIONS
Precision	0.654	0.810	0.821	0.892
F-Measure	0.703	0.818	0.820	0.876

Figure – 5.14: Comparison of Average Precision and F-measure of four semantic kernel functions



	SEMANTIC SMOOTHING KERNEL	LATENT SEMANTIC KERNEL	SEMANTIC WORDNET BASED KERNEL	KERNEL WITH IMPLICIT SUPERCONCEPT EXPANSIONS
MCC	0.355	0.631	0.641	0.758
ROC Area	0.718	0.867	0.901	0.917

Figure – 5.15: Comparison of MCC and RoC of four semantic kernel functions

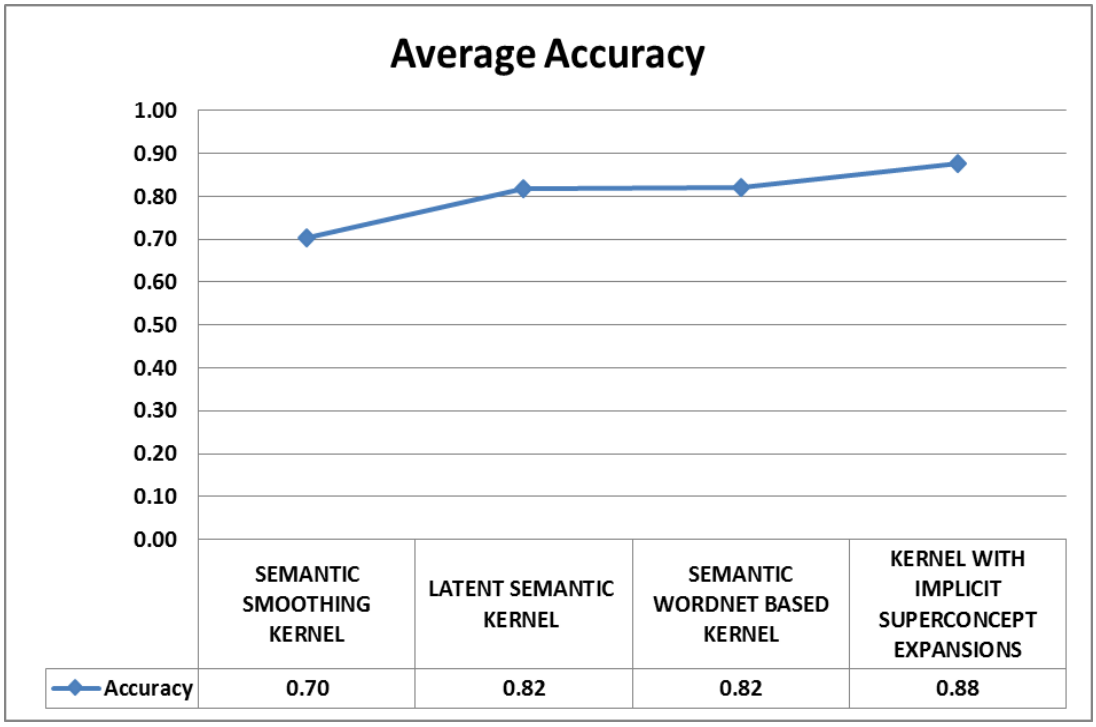


Figure – 5.16: Comparison of Accuracy of four semantic kernel functions

- Highest value of TP and the lowest value of FP makes kernel with implicit superconcept expansions more precise. This is correctly concluded when we observe the average values of precision and F-measure for all the three experimental setups.
- The kernel with implicit superconcept expansions clearly outplays rest of the kernel functions when accuracy of the classification system is calculated and compared. The important observation is that the accuracy of this kernel function is highest for all the three experimental setups implemented using different data sets.
- High value of MCC observed in case of kernel with implicit superconcept expansions reflects a strong

correlation between the observed and predicted values of binary classification.

- The observed values of AUC shows that kernel with implicit superconcept expansions and Semantic WordNet based Kernel have excellent values of RoC whereas Latent Semantic Kernel gives competitive values on a couple of occasions during the experimentation.
- Overall, average performance measures clearly reflect the superiority of “Kernel with implicit superconcept expansions” over the rest. It exhibits the highest accuracy with maximum RoC area coverage. Least value of FP-Rate and highest value of TP-Rate in case of this kernel demonstrate least alarm rate and high hit-rate respectively when compared with other semantic kernels.
- Performance and accuracy parameters for LSK and Semantic WordNet based kernel functions are comparable. Although both use different algorithms for classification, near equality among various evaluated parameters reflects a critical observation about their dependency on semantic knowledge sources. On one side, LSK uses internally extracted knowledge while other kernel uses a-priori external knowledge.

Four semantic kernel functions have been investigated with a focus on text classification. Similarity between terms, concepts and documents is computed considering semantic relationship between them. Performance parameters like TP Rate, FP Rate, Precision, Recall and F-measure are evaluated to investigate the reliability of four kernel functions. Other parameters like MCC, RoC Area and Accuracy provide an insight on how close the experimental results are with true values.

Semantic Smoothing Kernel is simple to implement and good in improving the performance of the classification system. But its implementation is very complex when number of support vectors handled by semantic proximity matrix are increased above few thousands. The complexity increases because of difficulty encountered while handling proximity matrix with increased size. Overall, this semantic kernel provides performance parameters which are lowest among all the evaluated kernels. But, being among the very first semantic kernels proposed, it provides a strong base to many semantic kernels that have been proposed thereafter. This kernel function finds a good fitment in all real life scenarios where domain knowledge is available a-priori and a quick but no so precise solution is required.

Latent Semantic Kernel can handle both text as well as non-text data which makes it more flexible in diverse scenarios. It is possible to apply this kernel irrespective of data original dimensionality. But complexity involved during performing eigenvalues decomposition poses a limitation on its usage. This kernel shows promising results when various performance parameters are evaluated. These values are comparable on a couple of instances to the second best kernel evaluated during the experimentation. But, when we test it on large imbalance data sets during text classification, it poses a threat to good performance. It finds an application in scenarios like non-text classification e.g. Radar data, satellite data, Ionosphere data classification. Also, it is suitable for classification of those data sets where a-priori knowledge is not available easily e.g. classification of medical documents.

Semantic WordNet-based Kernel provides a space which supports the similarity between terms of different surface forms based on external knowledge. It also helps in shunning the need of explicitly defining the term or sense clusters which introduce noise. In poor training data conditions, the WordNet prior knowledge can be effectively used to improve the accuracy of text classification. But there is a scope of improvement of the overall efficiency by exploring feature selection methods over the semantic kernel. Similar to semantic smoothing kernel, this kernel also finds its fitment in almost all real life scenarios.

Semantic Smoothing Kernel with implicit superconcept expansions has been observed with highest performance and accuracy among all the four semantic kernels. Performance is consistently good in those cases where little training data is available or the feature representations are extremely sparse. But, the overall performance of this kernel function can be further improved if a decent word sense disambiguation is used during the implementation [84]. This kernel function fits in scenarios where little training data is available or the feature representations are extremely sparse, for example, question answer classification.

Semantic kernel functions find their utility in many areas of text classification. Semantic Smoothing kernel using semantic knowledge from WordNet finds its application in almost all field of real life where domain knowledge is available a-priori. This gives us a quick solution in scenarios where we have ontologies available and we want to do some quick text analysis. An efficient and easy to

implement kernel function, semantic smoothing kernel is one of the most used semantic kernel function across the domain of text classification. Many variants of this kernel have been implemented to serve specific industry problems of textual analysis. Quite a few kernel functions have been implemented for scenarios where we have very less training data like question answer classification. Semantic smoothing kernel with implicit superconcept expansions, is a kernel function primarily address Q&A classification. Non-Text data is also one of the prime area where semantic kernel functions find their application. Latent Semantic Kernel offers a solution for classification of non-text data like images, radar and satellite data. This kernel also works well when we don't have prior domain knowledge available with us e.g. medical document classification. In current times, ontology enrichment and completing large ontologies are two sub-areas of text classification explored in great depth.

5.3 Result Comparison between Linear and Semantic kernel Implementations

The developed classification system has been tested using two scenarios, one being implemented using linear kernel functions and the other one implemented using semantic kernel functions.

Let us now compare and contrast two results obtained in chapter 4 and 5.

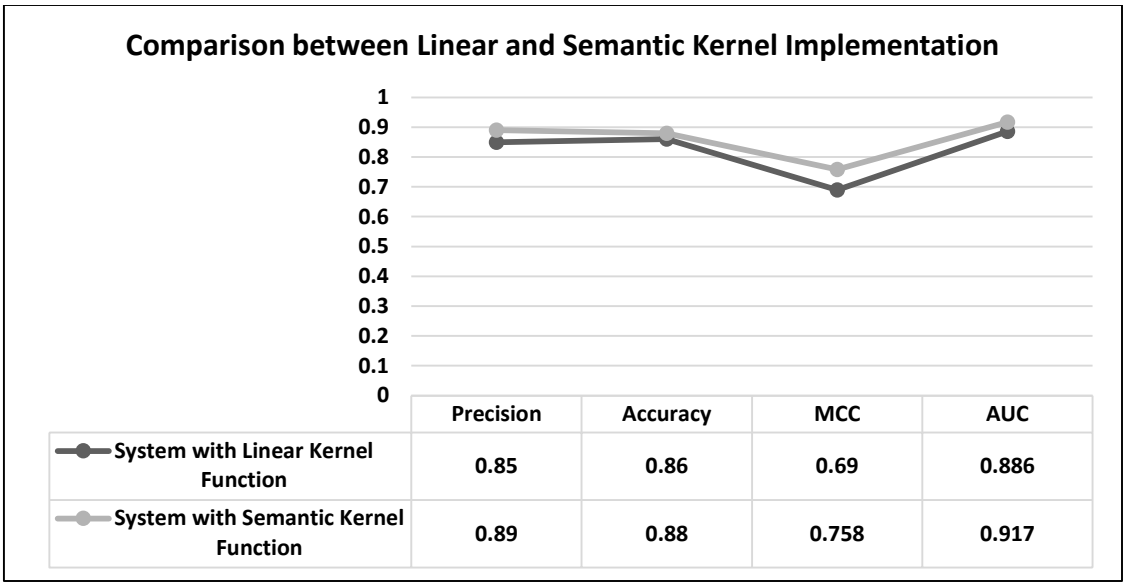


Figure – 5.17: System Comparison between Linear and Semantic Kernel Implementation

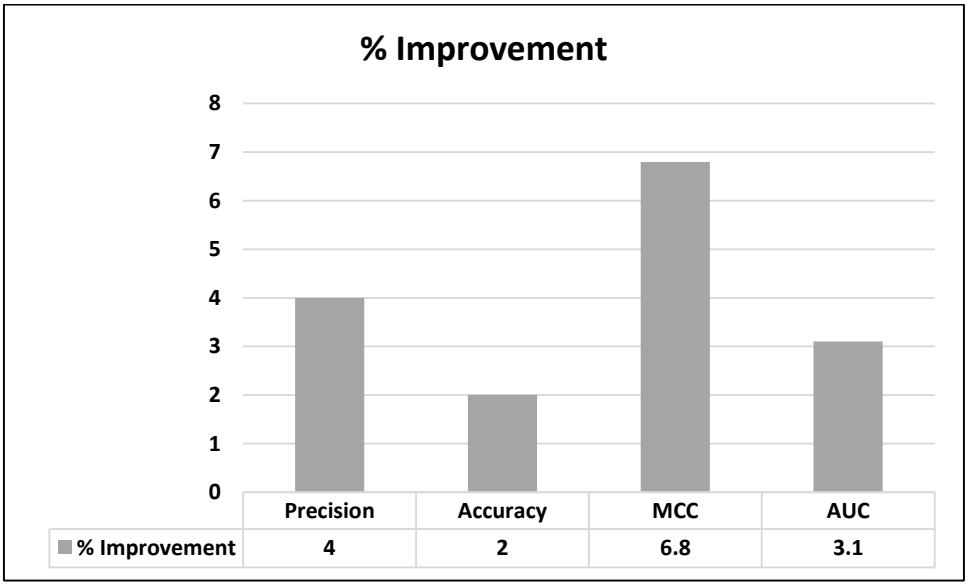


Figure – 5.18: Percentage Improvement when implemented using Semantic Kernel vis-à-vis Linear Kernel

Figure – 5.18 clearly shows the better utility of semantic kernels when compared with linear kernel in the context of our system implementation using SVM. Although all parameters

comparison is positive but there is definitely a scope of improvement as far as accuracy is concerned. This will formulate our future course of action.

Chapter – 6

Conclusions and Future Directions

6.1 Conclusions

In this research problem, we propose a self-governed ontology-based approach to classify unstructured textual documents purely in the relevant context of user query. The designed KDIMS system gets triggered when a user throws a query in a plain English language. Two branches of the system start working in parallel: one being the collection of relevant documents through focused web crawler and second being the build-up of seed ontology. Subsequent to data preprocessing and feature selection, again two processes start working in parallel: one being populating the domain ontology on top of seed ontology with the help of ontology manager, and second being the training of SVM based classifier. Towards the end, we have compared the two setups for this system and conclude that ontology based classifier shows promising results and performs better than the simple SVM based classifier. The whole system is implemented from scratch with no manually prepared seed ontology being used which is quite encouraging. We have also compared our framework with similar available frameworks and found a better usefulness of our framework in terms of self-governed learning system.

There are certain areas in the system design which point to our future work. First and the foremost is query manager which handles the query string fed by the user and converts

it into seed ontology by forming RDF relation graph. Fine tuning of this stage will provide better results. OWL may also be explored during automatic ontology population instead of RDF. Second area of improvement is feature extraction and selection. We need to explore more matured algorithm during this stage. It will be quite interesting to perform more in-depth analysis of semantic kernels with SVM instead of linear one [24, 25, 26].

Semantic kernel functions have been implemented to replace linear kernel functions in SVM in a bid to validate the KDIMS framework. This implementation helps us to explore a more rugged approach towards unstructured text classification. The role of semantic kernel functions in text classification has been investigated wherein similarity in given documents is evaluated by exploiting the semantic instead of syntactic relationships. Comparison of performance and accuracy parameters of four semantic kernel functions provides a direction about their usage in different scenarios of knowledge discovery from textual data. Experimental results demonstrate that out of the four evaluated kernel functions, semantic smoothing kernel with implicit superconcept expansions is the most reliable and accurate semantic kernel. LSK and Semantic WordNet based kernel gives almost similar performance and accuracy but both of them are distinct second. Semantic smoothing kernel is the last one in the comparison table, but it is not to be overlooked because this was the starting point in the field of semantic kernel functions.

A comparison between two KDIMS implementations, one using linear kernel function and the other using semantic kernel

function provides an insight into the usefulness of a-priori knowledge over syntax based classification. An overall improvement in various performance and accuracy parameters also validate our framework implementation with a-priori knowledge.

Analysis also provides us a direction that there is a scope of further improvement in terms of accuracy and performance of all the four semantic kernel functions. Many of the existing syntactic based kernel functions are matured enough to provide excellent performance and accuracy, but in the context of syntactic based knowledge discovery. Hence it will be good to investigate a combination of semantic kernel function with syntactic kernel function for text classification. It will also be quite interesting to explore these kernel functions on live web data which may test their utility in real business scenarios.

Therefore, we can conclude here that

- We have suggested a framework which may be used to extract information from unstructured data using semantic web mining standards (RDF)
- We have proposed a data model / Knowledge Driven Information management system (KDIMS) to manage the process of Information Extraction in the context of unstructured data.

The objectives of this research have been met as follows:

- We have designed a framework that will extract useful information from unstructured data using Semantic web mining standards.

- We have designed and developed a KDIMS which will interact with the documents to extract useful information using Semantic web mining standards.
- We have validated the developed model / system by comparing it with nearest available framework and also by incorporating semantic kernels for the implemented KDIMS.
- We have suggested business usefulness of this KDIMS to the organizations with a clear direction to use semantic kernel functions with SVMs along with system built ontology for unstructured text classification.

6.2 Future Directions

Rapid growth of WWW over the last couple of decades has resulted in a tsunami of data. With the advent of Big Data, it is challenging to understand and gain insight into this data when most of it is unstructured in nature. Gartner Inc. has identified “Big Data” and “Analytics” as two of the top 10 strategic technologies for 2012 and 2013 [35, 36]. This has brought a lot of research focus in the area of big data analytics in recent times. It will be quite interesting to explore semantic kernels while working on big data as the corpus. Currently, most of the semantic kernels perform reasonably well when the corpus size is small, say in gigabytes. How they will behave when the data size becomes peta-bytes and zeta-bytes? Also, how they will behave when number of features and concepts will increase multi-fold in this case, needs to be explored.

Social networking sites are one of the most important sources of big data generation in recent time. As we know, most of the data on these sites is nothing but short text

being inputted by users while communicating with fellow friends. We need to explore semantic kernels in the scenario of short text classification. Also, it needs to be seen when short text classification may be done with language independent semantic kernels.

Extracting feature representations suitable for machine learning algorithms from linguistic structures is typically difficult. This points to next level of exploration of semantic kernels in the context of machine learning algorithms. Also, as most of the semantic kernels explored in this survey exploited textual datasets. It will be critical to experiment the same semantic kernels when the data is non-textual type e.g. images, audio, video formats etc. Developing automatic vision algorithms to recognize tens of thousands or even millions of image categories will be an interesting field to be explored. Standard semantic role labeling task to include relations expressed by lexical items other than verbs and nominalizations is another field meant for future exploration.

Some of the open problems that can be explored in greater depth:

- The success of introducing a-priori semantic knowledge in text mining tasks depends on the strategic employment of word sense disambiguation (WSD) [84, 91]. It will be critical to observe their performance when a powerful WSD technique is applied.
- Combination of semantic kernels with tree kernels [22] has shown good performance as seen for Syntactic and Shallow

STK kernels [60]. Investigation can be done into the combination of semantic kernels with other types of kernels.

- Overall accuracy and performance parameters in all the eight kernels need a revisit. There is an ample scope for improvement in various parameters.
- Investigation of all these kernel functions with live web database may also be explored.

A few futuristic research areas in the field of text classification, need probing when implemented using semantic kernel functions:

- Behavior of semantic kernel functions when exploited in the context of Big Data Analytics.
- Application of these kernel functions for Short-Text classification while analyzing social networking sites during sentiment analysis.
- Non-text data analysis requires more exploration in the context of semantic kernel functions.
- Utility of semantic kernel functions while enriching the semantic knowledge for 3-D object classification needs probing. This exploration may lead to an automated robotic system that can efficiently detect and classify the objects in the environment.

Paper Published / Communicated

- Manuja, M., & Garg, D. (2011). Semantic web mining of un-structured data: Challenges and opportunities. International Journal of Engineering (IJE), 5(3), 268.
- Manuja, M., & Garg, D. (2013) Intelligent text classification system based on self-administered ontology. Online.journals.tubitak.gov.tr. DOI: 10.3906/elk-1305-112. [SCI Journal: Impact Factor 0.563] Available Online since 3-Sept-2013.
- Manuja, M., & Garg, D. Semantic Kernel Functions for Support Vector Machines – A Survey. Under Review in International Journal of Semantic Web and Information Systems (ISSN: 1552-6283). Science Citation Index Expanded with Impact Factor 2.308.
- Manuja, M., & Garg, D. Performance and Accuracy Analysis of Semantic Kernel Functions (PROG-04-2014-0028). Under Review in “Emerald - Program Electronic library and information systems” (ISSN: 0033-0337). Science Citation Index Expanded.

References

[1] A Maedche. (2002). “Ontology Learning for the Semantic Web”; Kluwer. ISBN: 0792376560

[2] Abe S. Support vector machines for pattern classification. New York: Springer-Verlag, 2005.

[3] Agirre, E., and Rigau, G. (1996), “Word sense disambiguation using conceptual density”, In Proceedings of the 16th conference on Computational linguistics-Volume 1, Association for Computational Linguistics, pp. 16-22.

[4] B. Berendt, A. Hotho, and G. Stumme. (2002). “Semantic Web Mining and the Representation, Analysis, and Evolution of Web Space”. Proceedings of the First International Semantic Web Conference on The Semantic Web. pp. 264 – 278

[5] Baader, F. (Ed.). (2003). The description logic handbook: theory, implementation, and applications. Cambridge university press.

[6] Basili, R., Cammisa, M. and Moschitti, A. (2005), “Effective use of WordNet semantics via kernel-based learning”, In Proceedings of the Ninth Conference on Computational Natural Language Learning, Association for Computational Linguistics, pp. 1-8.

[7] Belkin, M., Niyogi, P., & Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. The Journal of Machine Learning Research, 7, 2399-2434.

[8] Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. Scientific American, 284(5), 28-37.

[9] Bizer C, Heath T, Berners-Lee T. Linked Data—The Story so far. Int J Semant Web Inf 2009; 5: 1–22.

[10] Bloehdorn, S. and Sure, Y. (2007), “Kernel methods for mining instance data in ontologies”, In the Semantic Web, Springer Berlin Heidelberg, pp. 58-71.

[11] Bloehdorn, S., & Moschitti, A. (2007). Combined syntactic and semantic kernels for text classification. In Advances in Information Retrieval (pp. 307-318). Springer Berlin Heidelberg.

[12] Bloehdorn, S., Basili, R., Cammisa, M. and Moschitti, A. (2006), “Semantic kernels for text classification based on topological measures of feature similarity”, In Data Mining, 2006. ICDM'06. Sixth IEEE International Conference, pp. 808-812.

[13] Brank J, Mladenic D, Grobelnik M. Large-scale hierarchical text classification using SVM and coding matrices. In: Large-Scale Hierarchical Classification Workshop of ECIR 2010; 28 – 31 March 2010; Milton Keynes, UK.

[14] Buitelaar P, Cimiano P, Magnini B. Ontology Learning from Text: Methods, Evaluation and Applications. IOS Press, 2005.

[15] Campbell, C., & Ying, Y. (2011). Learning with support vector machines. Synthesis Lectures on Artificial Intelligence and Machine Learning, 5(1), 1-95.

[16] Chang C, Lin C. LIBSVM: a library for support vector machines. Software, 2001.

[17] Christopher CS, Tylman J. Enterprise Information Portals. Electron Libr 1998; 18: 354-362.

[18] Collins, M., & Duffy, N. (2001). Convolution kernels for natural language. In Advances in neural information processing systems (pp. 625-632).

[19] Craven M, DiPasquo D, Freitag D, McCallum A, Mitchell T, Nigam K, Slattery S. Learning to construct knowledge bases from the World Wide Web. Artif Intell 2000; 118: 69–114.

[20] Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines and other kernel-based learning methods. Cambridge university press.

[21] Cristianini, N., Shawe-Taylor, J. and Lodhi, H. (2002), “Latent semantic kernels”, Journal of Intelligent Information Systems, 18(2-3), pp. 127-152.

[22] Culotta, A., & Sorensen, J. (2004). Dependency tree kernels for relation extraction. In Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (p. 423). Association for Computational Linguistics.

[23] Cumby, C. and Roth, D. (2003), “On kernel methods for relational learning”, In ICML, pp. 107-114.

[24] D. Bitton, F. Faerber, L. Haas, J. Shanmugasundaram. (2006). “One platform for mining structured and unstructured data: dream or reality?”. Proceedings of the 32nd international conference on Very large data bases. pp. 1261 – 1262

[25] d’Amato, C. (2007). Similarity-based Learning Methods for the Semantic Web (Doctoral dissertation, PhD thesis, University of Bari).

[26] d’Amato, C., Fanizzi, N., & Esposito, F. (2008). Classification and retrieval through semantic kernels. In Knowledge-Based Intelligent Information and Engineering Systems (pp. 252-259). Springer Berlin Heidelberg.

[27] d’Amato, C., Fanizzi, N., Fazzinga, B., Gottlob, G., & Lukasiewicz, T. (2010b). Combining Semantic Web search with the power of inductive reasoning. In Scalable Uncertainty Management (pp. 137-150). Springer Berlin Heidelberg.

[28] d'Amato, C., Esposito, F., Fanizzi, N., Fazzinga, B., Gottlob, G., & Lukasiewicz, T. (2010a). Inductive reasoning and Semantic Web search. In Proceedings of the 2010 ACM Symposium on Applied Computing (pp. 1446-1447). ACM.

[29] d'Amato, C., Fanizzi, N., & Esposito, F. (2010c). Inductive learning for the semantic web: What does it buy?.Semantic Web, 1(1), 53-59.

[30] Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W. and Harshman, R. A. (1990), “Indexing by latent semantic analysis”, JASIS, 41(6), pp. 391-407.

[31] Dumais, S., Platt, J., Heckerman, D. and Sahami, M. (1998), “Inductive learning algorithms and representations for text categorization”, In Proceedings of the seventh ACM International Conference on Information and Knowledge Management, pp. 148-155.

[32] Fanizzi, N., & d’Amato, C. (2006). A Declarative Kernel for ALC Concept Descriptions. In Foundations of

Intelligent Systems (pp. 322-331). Springer Berlin Heidelberg.

[33] Fazzinga, B., Gottlob, G., Gianforme, G., & Lukasiewicz, T. (2008). From Web search to Semantic Web search. In InstitutfürInformationssysteme.

[34] Gamon, M. (2004). Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In Proceedings of the 20th international conference on Computational Linguistics (p. 841). Association for Computational Linguistics.

[35] Gartner Inc., "Gartner Identifies the Top 10 Strategic Technologies for 2012," 2011. [Online. Available: http://www.gartner.com/it/page.jsp?id=1826214]

[36] Gartner Inc., "Gartner Identifies the Top 10 Strategic Technologies for 2013," 2012. [Online. Available: http://www.gartner.com/it/page.jsp?id=2209615]

[37] Gärtner, T. (2003), “A survey of kernels for structured data”, ACM SIGKDD Explorations Newsletter, 5(1), pp. 49-58.

[38] Gärtner, T., Lloyd, J. W. and Flach, P. A. (2004), “Kernels and distances for structured data”, Machine Learning, 57(3), pp. 205-232.

[39] Gärtner, T., Lloyd, J. W., & Flach, P. A. (2003). Kernels for structured data (pp. 66-83). Springer Berlin Heidelberg.

[40] Golub, G. H. and Reinsch, C. (1970), “Singular value decomposition and least squares solutions”, Numerische Mathematik, 14(5), pp. 403-420.

[41] Gupta V, Lehal G. A Survey of Text Mining Techniques and Applications. Journal of Emerging Technologies in Web Intelligence 2009; 1: 60-76.

[42] Heath T, Bizer C. Linked Data: Evolving the Web into a Global Data Space. Morgan & Claypool Publishers, 2011.

[43] Hougardy, S. (2010), “The Floyd–Warshall algorithm on graphs with negative cycles”, Information Processing Letters, 110(8), pp. 279-281.

[44] J. Han, M. Kamber. (2001) “Data Mining Concepts and Techniques”. Academic Press, Morgan Kaufmarm Publishers. ISBN 1-55860-489-8.

[45] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features (pp. 137-142). Springer Berlin Heidelberg.

[46] Joachims, T. (1999), “Making large-Scale SVM Learning Practical”, Advances in Kernel Methods: Support Vector Learning, B. Schölkopf, C. Burges and A. Smola (ed.), MIT-Press.

[47] Kohavi, R. (1995), “A study of cross-validation and bootstrap for accuracy estimation and model selection”, In IJCAI, Vol. 14, No. 2, pp. 1137-1145.

[48] L. Dey , S. K. M. Haque. (2009, July). “Studying the effects of noisy text on text mining applications”. Proceedings of The Third Workshop on Analytics for Noisy Unstructured Text Data. Barcelona, Spain

[49] Lang, K. NewsWeeder: learning to filter netnews. In: 12th International Conference on Machine Learning; 9 – 12 July 1995; Lake Tahoe, US: IMLS. pp. 331–339.

Data set accessed from http://qwone.com/~jason/20Newsgroups/.

[50] Lewis D. The Reuters-21578 text categorization test collection, 1997.

[51] Li, X., & Roth, D. (2002). Learning question classifiers. In Proceedings of the 19th international conference on Computational linguistics-Volume 1 (pp. 1-7). Association for Computational Linguistics.

[52] Luong HP, Gauch S, Wang Q. Ontology Learning Through Focused Crawling and Information Extraction. In: International Conference on Knowledge and Systems Engineering; 13 - 17 October 2009; Hanoi: IEEE Computer Society. pp. 106–112.

[53] M. Niepert, C. Buckner, J. Murdock, C. Allen. (2008). "InPhO: a system for collaboratively populating and extending a dynamic ontology". Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries, Pittsburgh PA, PA, USA. pp. 429-429

[54] M. Rajman, R. Besancon. (1997). "Text Mining - Knowledge extraction from unstructured textual data". In Proceedings of the 7th IFIP Working Conference on Database Semantics (DS-7). pp. 7-10

[55] Maedche A, Staab S. Ontology Learning for the Semantic Web. IEEE Intell Syst 2001; 16: 72-79.

[56] Mahoui, M., Bhargava, B., & Mohania, M. (2001, June). Data Mining For Web Security: UserWatcher. In *Proc of the 2001 International Conference on Internet Computing*.

[57] Matthews, B. W. (1975), "Comparison of the predicted and observed secondary structure of T4 phage

lysozyme", Biochimica et Biophysica Acta (BBA)-Protein Structure, 405(2), pp. 442-451.

[58] Mavroeidis, D., Tsatsaronis, G., Vazirgiannis, M., Theobald, M., & Weikum, G. (2005). Word sense disambiguation for exploiting hierarchical thesauri in text classification. In Knowledge Discovery in Databases: PKDD 2005 (pp. 181-192). Springer Berlin Heidelberg.

[59] Miller, G. A. (1995), "WordNet: a lexical database for English", Communications of the ACM, 38(11), pp. 39-41.

[60] Moschitti, A., Quarteroni, S., Basili, R., & Manandhar, S. (2007). Exploiting syntactic and shallow semantic kernels for question answer classification. In ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS (Vol. 45, No. 1, p. 776).

[61] Muller, K. R., Mika, S., Ratsch, G., Tsuda, K. and Schölkopf, B. (2001), "An introduction to kernel-based learning algorithms", Neural Networks, IEEE Transactions on, 12(2), pp. 181-201.

[62] Nadaraya, E. A. (1964). On estimating regression. Theory of Probability & Its Applications, 9(1), 141-142.

[63] Navigli R, Faralli S, Soroa A, de Lacalle OL, Agirre E. Two birds with one stone: learning semantic models for Text Categorization and Word Sense Disambiguation. In: 20th ACM Conference on Information and Knowledge Management; 24 - 28 October 2011; Glasgow, United Kingdom. pp. 2317–2320.

[64] Navigli, R., Velardi, P., & Gangemi, A. (2003). Ontology learning and its application to automated terminology translation. Intelligent Systems, IEEE, 18(1), 22-31.

[65] Pallottino, S. (1984), “Shortest-path methods: Complexity, interrelations and new propositions”, Networks, 14(2), pp. 257-267.

[66] Pang, B. and Lee, L. (2004), “A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts”, In Proceedings of the 42nd annual meeting on Association for Computational Linguistics, Association for Computational Linguistics, pp. 271.

[67] R. Ghani and Carlos. (2006, December). “Data mining for business applications”. KDD-2006 workshop. Volume 8, Issue 2. pp. 79 – 81

[68] R. J. Mooney, R. Bunescu. (2005). “Mining knowledge from text using information extraction”; ACM SIGKDD Explorations Newsletter. pp. 3 – 10

[69] RDF Working Group. Resource description framework (RDF); W3C-Semantic Web, 2004.

[70] Resource Description Framework (RDF) Schema Specification. (2000) In W3C Recommendation.

[71] Reuters Collection (1995), http://www.daviddlewis.com/resources/testcollections/reuters21578/ (accessed 25 April 2014).

[72] Salton G, Buckley C. Term-weighting approaches in automatic text retrieval. Inform Process Manag 1998; 24: 513–523.

[73] Schmidt-Schauß, M., & Smolka, G. (1991). Attributive concept descriptions with complements. Artificial intelligence, 48(1), 1-26.

[74] Schölkopf, B. and Smola, A. J. (2002), “Learning with kernels”, The MIT Press, pp. 366-369.

[75] Schölkopf, B., Burges, C. J. and Smola, A. J. (Eds.) (1999), Advances in kernel methods: support vector learning, The MIT press.

[76] Schölkopf, B., Smola, A., & Müller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. Neural computation, 10(5), 1299-1319.

[77] Sebastiani, F. (2002). Machine learning in automated text categorization. ACM computing surveys (CSUR), 34(1), 1-47.

[78] Shawe-Taylor, J. and Cristianini, N. (2004), “Kernel methods for pattern analysis”, Cambridge university press.

[79] Sheth A. Computing for human experience: Semantics-empowered sensors, services, and social computing on the ubiquitous Web. IEEE Internet Comput 2010; 14: 88-91.

[80] Singh, M., Singh, S., & Gupta, S. (2014). An information fusion based method for liver classification using texture analysis of ultrasound images. *Information Fusion*, *19*, 91-96.

[81] Siolas, G. and d'Alché-Buc, F. (2000), “Support vector machines based on a semantic kernel for text categorization”, In Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference, Vol. 5, pp. 205-209.

[82] Soman, K. P., Loganathan, R., & Ajay, V. (2009). Machine Learning with SVM and other Kernel methods. PHI, India.

[83] Speretta M, Gauch S. Using text mining to enrich the vocabulary of domain ontologies. In: IEEE International Conference on Web Intelligence and Intelligent Agent Technology; 9 – 12 December 2008; Sydney, Australia: IEEE. pp. 549–552.

[84] Sreedhar, J., Raju, S. V., Babu, A. V., Shaik, A., & Kumar, P. P. Word Sense Disambiguation: An Empirical Survey. *International Journal of Soft Computing and Engineering (IJSCE) ISSN*, 2231-2307.

[85] Steinwart, I., & Christmann, A. (2008). Support vector machines. Springer.

[86] Sujatha, B., Raju, D. S. V., & Shaziya, H. (2012). A Survey of Natural Language Interface to Database Management System. International Journal of Science and Advanced Technology, ISSN, 2221-8386.

[87] Tim Berners-Lee. "Semantic Web Roadmap". http://www.W3.org/

[88] Thampi, S. M. (2008). An Introduction to Knowledge Management. *arXiv preprint arXiv:0812.0438*.

[89] Tong, S. and Koller, D. (2002), "Support vector machine active learning with applications to text classification", The Journal of Machine Learning Research, 2, pp. 45-66.

[90] Vapnik, V., Golowich, S. E. and Smola, A. (1997), "Support vector method for function approximation, regression estimation, and signal processing",

Advances in neural information processing systems, pp. 281-287.

[91] W. Fan, L. Wallace, S. Rich, Z. Zhang. (2006, September). “Tapping the power of text mining”. Communications of the ACM. Volume 49, Issue 9. pp. 76 – 82

[92] Wang P, Domeniconi C. Building Semantic Kernels for text classification using Wikipedia. In: 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; August 24 – 27, 2008; Nevada, Las Vegas. New York, NY: ACM Press. pp. 713-721.

[93] Watson, G. S. (1964). Smooth regression analysis. Sankhyā: The Indian Journal of Statistics, Series A, 359-372.

[94] Wei, C. P., Piramuthu, S., & Shaw, M. J. (2003). Knowledge discovery and data mining. In *Handbook on Knowledge Management* (pp. 157-189). Springer Berlin Heidelberg.

[95] Wei GY, Wu GX, Gu YY, Ling Y. An Ontology Based Approach for Chinese Web Texts Classification. Inform Technol J 2008; 7: 796-801.

[96] Witten, I. H., & Frank, E. (2005), Data Mining: Practical machine learning tools and techniques, Morgan Kaufmann.

[97] Zelenko, D., Aone, C. and Richardella, A. (2003), “Kernel methods for relation extraction”, The Journal of Machine Learning Research, 3, pp. 1083-1106.

[98] Agrawal R., Grosky W, Fotouhi F, & Wu C (2007). Application of diffusion kernel in multimodal image retrieval. IEEE intl Symposium, pp 271-276, IEEE 2007.