Intelligent text classification system based on self-administered ontology

Manoj MANUJA[1,2], Deepak GARG[1]

[1] Computer Science & Engineering Dept., Thapar University, Patiala, India

[2] Education and Research Dept., Infosys Ltd., Chandigarh, India

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 share the details of a proof of concept of 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.

Keywords

Ontology, Support Vector Machine, Resource Description Framework, Text Classification.

1. Introduction

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 [1]. 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 [2]. 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.

A lot of work has been done on developing semantic based classification systems during the past few years [3,4,5]. 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 [6,7]. Ontologies provide a controlled vocabulary of concepts along with their relations explicitly defined using machine process-able semantics. 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 [8]. Quite a few frameworks are available which address this automation problem as mentioned in next section where related work in this field is discussed. Most of them use full-fledged pre-built domain or seed ontology to start with the automation and enrichment process.

In this paper, we share the details of a proof of concept (PoC) of a 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 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.

The paper is organized as follow. Section 2 provides the related work in this field. Section 3 provides a complete view of the system framework. Section 4 gives the insight on the experimental results achieved during the implementation with section 5 provides a detailed analysis. Comparison of our framework with similar closest frameworks is done in section 6. Section 7 concludes the paper with section 8 provides the references being used.

2. Related Work

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

Maedche and Staab [9] 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.

Navigli et al. [10] 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.

Wei et al. [11] 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.

Luong et al. [12] suggested a framework for ontology learning that uses a web crawler to retrieve documents from web, identifying domain specific documents using SVM (Support Vector Machine) 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.

Brank et al. [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.

Speretta and Gauch [14] 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.

3. System Framework

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 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.

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.

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

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 [15]. 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.

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 [16]. 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.

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 [17]. 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 given 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 >

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.

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 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.

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.

4. Experimental Results

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. We have done experiments with below mentioned two setups using LIBSVM package [18]:

- SVM based classifier with linear kernel
- SVM based classifier with ontology based approach

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 [19].

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 = P / (P + S)

Recall = Number of correctly identified items as percentage of the total number of correct items = P / (P + R)

F-measure = Weighted average of precision and recall = (2 * Precision * Recall) / (Precision + Recall)

Accuracy = degree of conformity of a measured quantity to its true value = (P + Q) / T

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. 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.

MCC (Matthews correlation coefficient) is a correlation coefficient between the observed and predictive binary classification, 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 = P / (P + S)

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

Use-Case1: Offline classification:

Three benchmark datasets namely Reuters-21578 [21] (Dataset available at http://www.daviddlewis.com/resources/testcollections/reuters21578/) , 20-Newsgroups collection [22] (Dataset available at http://qwone.com/~jason/20Newsgroups/) , and WebKB [23] (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.

It is clear from Table – 1 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.

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.

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.

The experimental results are quite heartening. It is clear from Table – 3 that SVM with system built ontology classifier is able to provide slightly better results than simple SVM classifier. Although average accuracy is improved by 0.3 factor, average F-measure improvement is just 0.1.

5. 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 3. Average MCC values calculated from various performance parameters are as shown in Table 4. All positive values of MCC (> 0) depict a reasonable representation of quality predictions as received from the experimental setup.

High accuracy of the system is also evident from the convex RoC under AUC analysis of two classifiers as shown in Figure 4 (usecase1) and Figure 5 (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.

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.

6. Comparison with other methodologies

Our framework is similar to the techniques such as suggested by Luong et al. [12] 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 [12] 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 [12] 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 [12] 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 [12] 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 [12] 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 [12] have used a threshold of ‘0.6’ during the experimental set-up, our framework uses a threshold of ‘1’. This implies that [12] 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 [12] (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.

7. Conclusion and future work

In this PoC, we propose a self-governed ontology-based approach to classify documents purely in the relevant context of user query. The whole 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 the experiment using semantic kernels with SVM instead of linear one [24, 25, 26].

8. References

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

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

[3] Sebastiani F. Machine learning in automated text categorization. Comput Surv 2002; 34: 1-47.

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

[5] 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.

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

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

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

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

[10] Navigli R, Velardi P, Gangemi A. Ontology Learning and Its Application to Automated Terminology Translation. IEEE Intell Syst 2003; 18: 22-31.

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

[12] 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.

[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] 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.

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

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

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

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

[19] Kohavi, Ron. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: International Joint Conference on Artificial Intelligence - IJCAI; 1995; Montréal, Québec, Canada. pp. 1137-1145.

[20] Matthews Brian W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. BIOCHIM BIOPHYS ACTA 1975; 405: 442-451.

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

[22] 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.

[23] 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.

[24] Siolas G, d'Alche Buc F. Support vector machines based on semantic kernel for text categorization. In: International Joint Conference on Neural Networks; 27 July 2000; Como, Italy: IEEE. pp. 205–209.

[25] Cristianini N, Shawe-Taylor J, Lodhi H. Latent Semantic Kernels, J Intell Inf Syst 2002; 18: 127-152.

[26] 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.

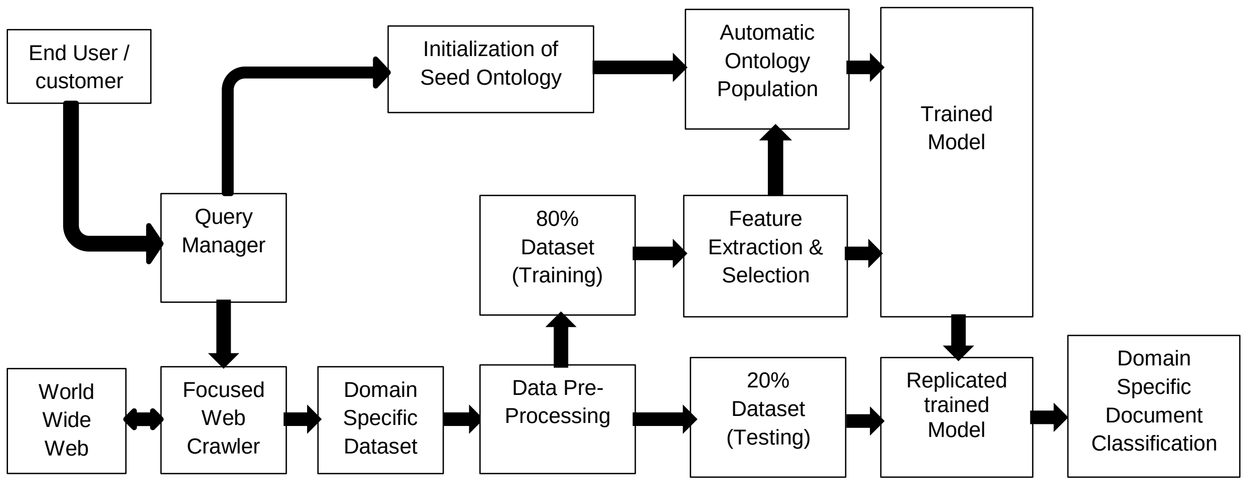

Figure 1: System Design of Self-governed Ontology based classification system

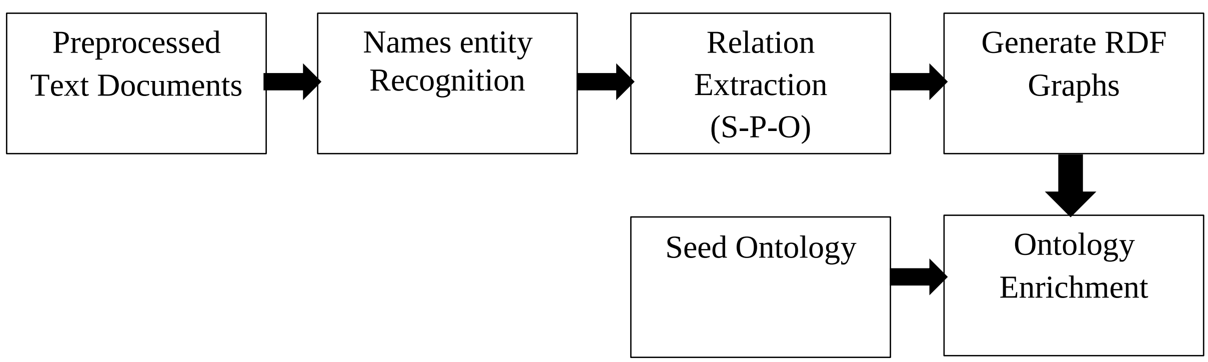

Figure 2: Process of Ontology population by Ontology Manager

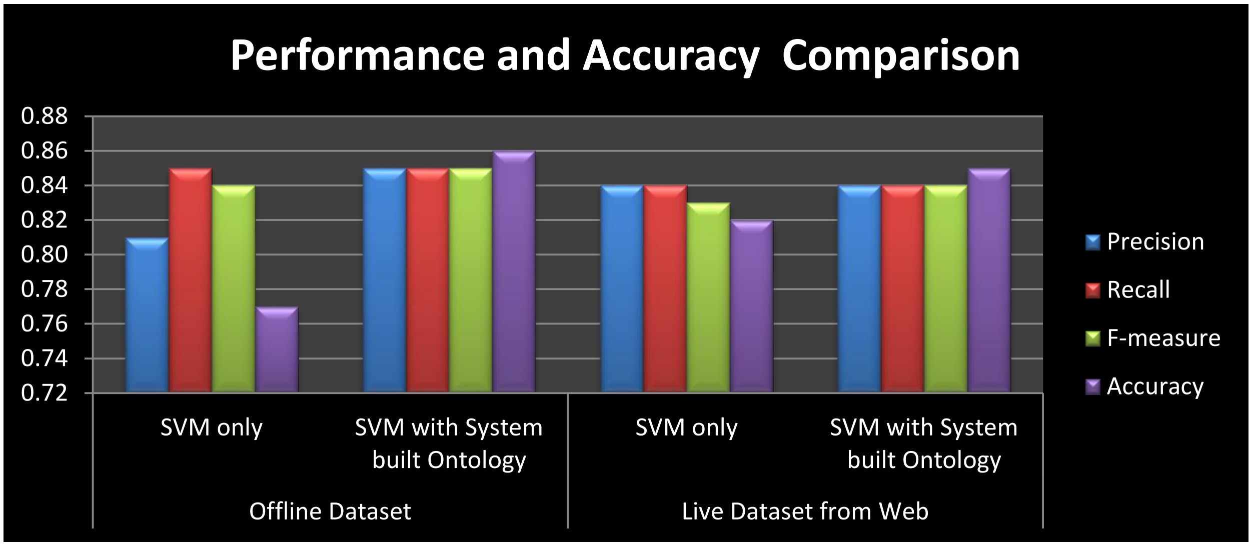

Figure 3: Comparison of performance and accuracy of two classifiers

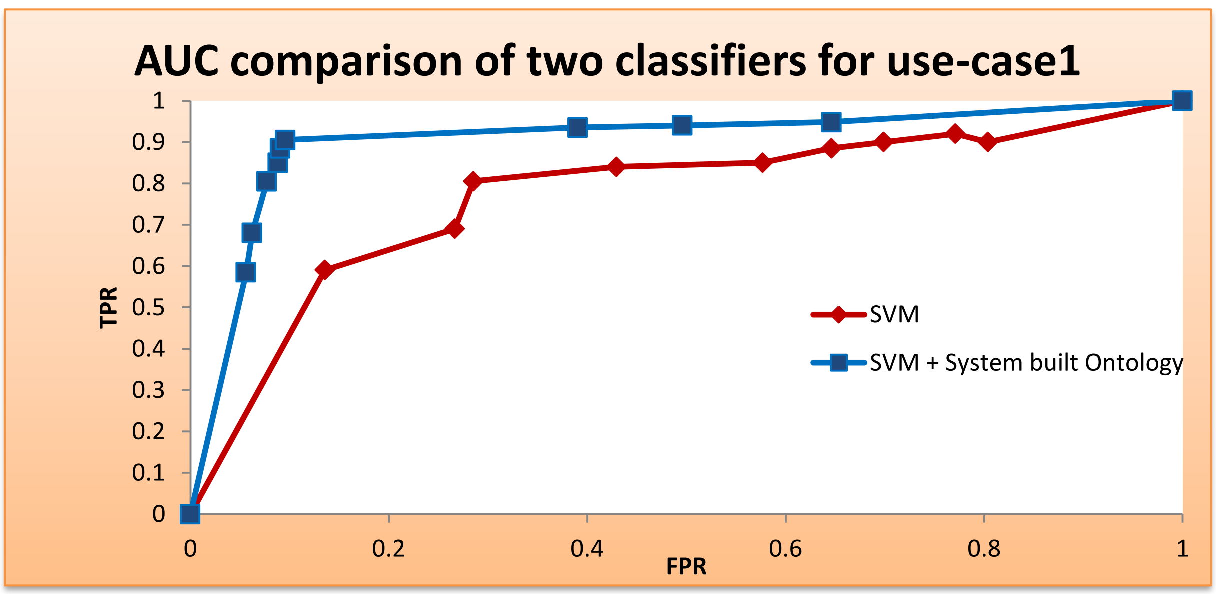

Figure 4: AUC Comparison of two classifiers for use-case1

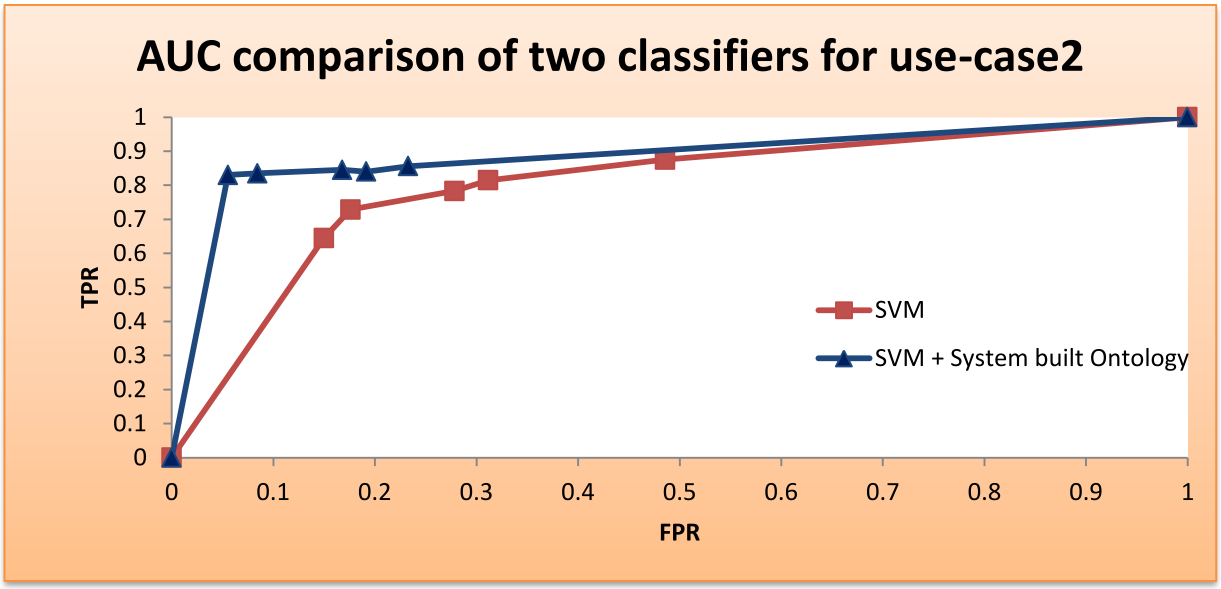

Figure 5: AUC Comparison of two classifiers for use-case2

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 1: Performance metrics and Accuracy of 3 Benchmark DBs under User-Case1

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 2: User Query response under Use-Case2

	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 3: Performance metrics under use-case2

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

Table 4: MCC metrics