RC25085 (W1012-092) December 21, 2010
Computer Science

IBM Research Report

Fuzziness Reduction and Training Set Generation for Text Classification in IT Services Domain

Yu Deng, Ruchi Mahindru, Nithya Rajamani, Soumitra Sarkar, Rafah Hosn, Murthy Devarakonda
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

Fuzziness Reduction and Training Set Generation for Text Classification in IT Services Domain

Yu Deng, Ruchi Mahindru, Nithya Rajamani, Soumitra Sarkar, Rafah Hosn, Murthy Devarakonda
IBM T.J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Abstract

Extracting domain-specific concepts from a large set of documents containing unstructured text to enable concept-based search has become a critical step for fulfilling the core information management requirements in enterprises today. Given the demanding computational requirements of performing text analysis for concept extraction, and the small fraction of documents that typically have useful concepts worth extracting (less than 1% in our system), a key challenge to be addressed is how to perform an intelligent selection of documents for further detailed analysis without compromising recall.

In this paper, we present a methodology to semi-automatically generate training sets for a classifier and an algorithm to reduce the ambiguity inherent in our data, for using a Support Vector Machine (SVM) classifier to identify candidate documents for concept extraction in the IT Services domain. In such proprietary domains, due to privacy and security concerns, very few training datasets are publicly available to train a classifier. Furthermore, documents in the domain are normally noisy and fuzzy (contain terms not discriminating across categories). Experiments show that our method has improved the SVM performance, achieving an accuracy of 92%, a recall of 99.2% and an F-measure of 92.6%.

1 Introduction

Several recent studies [10, 4, 2] targeting the business community (particularly sales productivity and effectiveness) have indicated that there is a critical need for convergence of both process and information enablers within the enterprise. Sales enablement is a particularly hot topic in Knowledge Management; it is about empowering the sales practitioners by providing them information pertaining to their everyday jobs. Our solution follows this best-practice paradigm and is targeted towards meeting the information needs of sales professionals, in the context of the business processes (sales process) they are engaged in. Traditional search solutions, which return a list of documents in response to a query, have limited scope and cannot sufficiently help professionals who work in an information-based "knowledge enterprise". We built a solution and tool, *Enterprise Information Leverage* (EIL) [23, 7], that goes beyond search and enables knowledge within the context of sales process for a community of practice - specifically, the sales and solutioning practitioners involved in selling IT services.

The information requirements that can help these practitioners win IT service opportunities (also referred to as deals or engagements) are reasonably well defined [7]. The practitioners typically need information from past deals in order to leverage the innovative aspects, strategies and lessons learned in their current engagements. Their information requirements can be categorized into a set of semantic concepts, which the EIL search platform extracts from unstructured documents. These unstructured documents have been created as a part of the sales process in past deals [7]. That targeted concept extraction goes through a cleansing process to remove any sensitive customer information.

Some of the semantics concepts that EIL currently extracts are:

- Win Strategy: the broad strategy that the sales team use to approach and solve the customer's business problems in order to deliver a winning proposition for a deal.
- Technology Solutions: the IT services solutions (e.g., server management, network management, etc.) that are proposed to the customer associated with a specific deal.
- Social Networking: the people who work on a deal,

their roles and expertise (e.g., solution architect - help desk, sales executive, and so on).

Such targeted information extracted by the EIL search system can help IT sales specialists leverage the strategies used in previous deals, especially where the customer pain points, their requirements for technology solutions, their industry and the deal complexity match their current opportunity. In this paper, we focus on the win strategy concept. For detailed information on the other concepts, please see [7, 23].

The concept of a win strategy is complex. It consists of many sub-concepts such as the use of innovation themes, use of a technical solution that provides compelling value-add to the customer's current environment, or the use of competitive pricing or financing models as a strategy to win the deal. It is not trivial to mine such complex concepts from unstructured text, as summarized in [5], especially when only a small fraction of documents have useful concepts worth extracting (e.g., less than 1% documents in our system contain the win strategy concept). Therefore, one of the key challenges is how to efficiently identify the right candidate documents for more rigorous analysis.

We present the use of a supervised text classification technique, Support Vector Machines (SVM) [16, 17], to address the challenge – to use binary text classification to determine whether a document has good win strategy content. Text classification is a technique used to automatically determine the category that a document (or part of a document) belongs to, based on the particular topic or characteristics of interest that the document contains. Supervised classification methods rely on pre-labeled data to train the model, and then use the model to classify new incoming data.

SVM has been widely used in many domains, such as Bioinformatics [15] and news data [26]. However, to the best of our knowledge, there is no published work describing the application of this technique to documents which contain concepts from the IT services domain. In such proprietary domains, due to customer related constraints, very few training sets are publicly available. Yet, manually labeling documents is a time-consuming and tedious task. This is one of the major obstacles to applying supervised classification techniques to these domains, compared to other domains such as news data, where people have spent a lot of time and effort on labeling documents to generate benchmarks.

Another challenge in the IT services domain is the fuzziness of the data. We highlight several aspects of this challenge in Section 2.1. Besides the lack of structure and standardization of engagement documents, the concepts we need to mine are very broad and abstract, which means that there is a high intersection between the terms used in documents that contain relevant concepts and those that do not. In addition, template documents and general discussions about the concepts are often in the corpus, which should not be considered as good matches.

In this paper, we present our work on handling these challenges when using SVM to identify candidates for concept extraction in the IT services domain. Figure 1 shows a high level view of concept extraction, which sets the context of our work. The component of document selection is the focus of this paper. When a document passes through that component, it goes into the detailed content analysis for extracting information related to the concepts. This two-step process is the major reason why we need to achieve high recall for document selection.

In Section 2, we give more details on these challenges. In Section 3, we discuss our methodology to semi-automatically generate a training set by leveraging domain knowledge. This method expedites the process of selecting good versus bad examples and significantly reduces the human effort involved in the process. We have leveraged this methodology to quickly extract an adequate training set from a pool of 600,000 documents. In Section 4, we present our technique on reducing fuzziness in the feature vector representation of a document. This technique has significantly improved the performance of the SVM classifier, especially the recall which is important for document selection. We then show experimental results in Section 5 and discuss related work in Section 6. We conclude in Section 7.

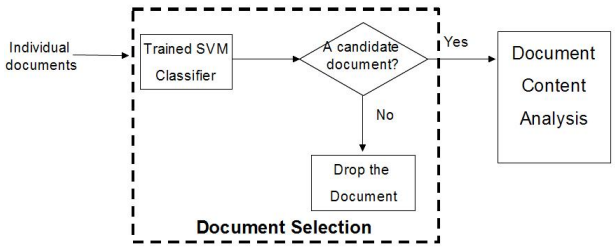

Figure 1. High Level View of Concept Extraction

2 Background Information

2.1 Characteristics of Engagement Documents

The documents analyzed by the EIL search tool are produced during the initial phase of an IT services deal, before a contract is signed. During this phase, referred to as *Engagement*, a team of specialists spend several weeks, sometimes months, at a client location, understanding the

Request for Proposal (RFP), analyzing the customer environment, creating, sizing and costing the overall solution to take over management of various aspects of the customer's IT environment.

Various types of documents are created during the Engagement phase. Documents created after an initial understanding of the customer pain points and requirements, and the competition involved, should include a description of the strategy that the team will use to win the deal (i.e., win strategy documents). Technical solution documents as well as the RFP response describe the services to be offered, and the final presentation to customer executives should describe the technical, financial and other advantages to be gained if the team win the deal. These documents normally contain customer specific languages and formats, which introduce noise and heterogeneity issues.

There are several challenges in mining documents produced as a byproduct of this selling process. First, the Engagement team work under heavy time pressure and therefore cannot create reusable high quality collateral devoid of customer sensitive information. The majority of the documents produced during this phase of the business process are not peer reviewed and edited for style and content, as technical papers or newspaper articles typically are. We call this characteristic "feature variation", i.e, each feature can have various forms; for example, win-strategy, win/strategy and win-win strategy are different forms of the feature "win strategy".

Second, the semantic concepts that the EIL tool attempts to mine are not as well-defined as the ontology for a certain class of news stories, or a branch of medicine or medical research, typically is. For example, the concept of a win strategy is very broad in scope and the vocabulary (terminology) used to describe such strategies can vary widely across documents produced by different teams spread across several continents. As such, there are many aspects of a win strategy that can be used to guide the Engagement team's approach to formulating the solution. In addition, there is a high intersection between the terms used in documents that contain relevant win strategy concepts and those that do not. We refer to this characteristic as "domain fuzziness".

Third, inferring semantics from a piece of text to determine that it contains a useful description of a win strategy can be difficult. On one hand, win strategy can be mentioned in many different types of documents. Due to the heterogeneity of the documents in IT services domain, we cannot expect structural coherence among them. On the other hand, the lack of structure in the documents makes the extraction not easy. For example, in some documents, specific descriptions are grouped into multiple "sections" where the only indication of a section heading is the title occurring on a line by itself.

Finally, win strategies documented in a "generic" language, simply extolling the reputation or experience of the service provider, are less useful than strategies that lay out in detail what specific approaches (technical, management, governance, etc.) the team will undertake in delivering the services to beat the competition. Using text analysis techniques to distinguish generic descriptions from specific (more valuable) ones can be challenging. Furthermore, those documents containing templates or sample information describing how to formulate a win strategy use the same vocabulary as the good win strategy documents themselves, which makes the task of extraction all the more difficult.

2.2 SVM Classifier

The Support Vector Machines (SVM) technique, introduced by Vapnik [24], has been widely used in many domains for text classification tasks [16] [17] [9] [22]. The main idea of SVM is to find a separating hyperplane that splits positive examples from negative examples with the largest distance between the hyperplane and the two example sets.

Recent studies have shown that SVM classifiers work well for text categorization and often outperform other classifiers [16] [26] [22]. SVM classifiers are robust with regard to the dimensionality of the feature space. In addition, they are easy to train and fast to run. It is indeed these characteristics that led us to choose an SVM classifier to filter documents for the purpose of concept extraction.

To apply the SVM method, each document should be represented as a vector. Normally, each word in the vocabulary of the corpus becomes a dimension (feature) of a vector. Sometimes, domain specific features are selected to improve the performance of SVM. In Section 5, we present the features used in our experiments and describe how we have used domain knowledge for feature selection.

3 Semi-automatic Training Set Generation

It is tedious and time-consuming to manually mark good/bad documents for generating a training set. In some domains where intensive studies have been done, e.g., the domain of news articles, people have spent a huge amount of time and effort to define benchmarks. But in new domains such as IT services, where benchmarks are not available, preparing a training set can be an impediment to applying supervised learning techniques. We address this problem by proposing a semi-automatic methodology to generate a training set. Figure 2 illustrates the main algorithm behind this methodology.

The domain knowledge "module" operates in the Unstructured Information Management Architecture (UIMA) [11] environment. In line 3, the set of regular expressions

SEMI-AUTOMATIC GENERATION OF TRAINING SET
/* Input: A set S of documents */
/* Output: A set of good examples from S and a set of bad examples from S */
1) for each document in S
2) Run the UIMA annotator pipeline to create positive sentiment annotations;
3) Run the UIMA annotator pipeline to create annotations for words or phrases in the document that match the set of regular expressions which encode domain knowledge;
4) Compute a set of metrics (including normalized score normS and number of win strategy features wsFeatures) based on the annotations;
5) if (normS $\geq T_1$ && wsFeatures $\geq T_2$)
6) Mark the document as a good example;
7) else if (normS $\leq T_3$ && wsFeatures $\leq T_4$)
8) Mark the document as a bad example;
9) Output the results for manual validation;

Figure 2. Algorithm for Domain Knowledge Based Semi-automatic Generation of Training Set

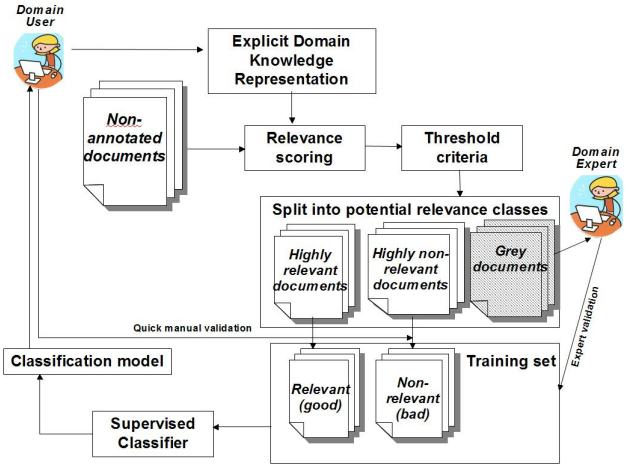

Figure 3. Architecture for Domain Knowledge Based Semi-automatic Generation of Training Set

is one of the key components in the algorithm. We have encoded our domain knowledge as text patterns representing concepts in these expressions. They are split into multiple levels of patterns – from expressions that very much guarantee the presence of a concept, expressions that are sub-concepts of the concept to be extracted, to expressions that signify discussions of the concept. For example, one of the expressions to capture pricing information is as follows:

(?i)((.*(competitive|differentia|compel).*(pric(ing|e)|cost|financ).*)|(.*(pric(ing|e)|cost|financ).*(strateg|model|saving|reduction|approach).*))

These expressions are given different importance levels to calculate the final relevance score. Every document in the data set is fed to the Relevance Scoring component, as shown in Figure 3, which highlights the interaction between the main components in this methodology. The Relevance Scoring component comes up with a relevance score based on how many times the different expressions appeared and the context they appeared in (e.g., in the title or section headings of a document versus the body).

In line 4, a set of metrics (including the relevance score) are computed based on the positive sentiment annotations (returned by an analysis engine in the UIMA "library"), as well as words and phrases identified in line 2 and 3. These metrics include the coverage of the win strategy concept within a document, the number of win strategy features and the number of positive features (those features indicate discussion of the win strategy concept). We assign different weights to those features based on their importance to the concept of interest and compute the relevance score based on the weighted features. We then normalize the score using the length of the document to obtain a normalized document score, which is normS in the algorithm.

In lines 5 and 7, the parameters T_1, T_2, T_3 and T_4 are thresholds for selecting documents. The last step (line 9) of the algorithm outputs results for human validation. Since we have a set of annotations produced for each returned document, it is easy for a human expert to validate if a document is good or bad by looking at its annotations. Without these annotations, a human expert would have to read the document thoroughly without any navigational aid.

We applied this methodology to generate a training set for the SVM classifier. In our setting, a document is labeled *good* if it contains relevant concepts that are worth further analysis and extraction, whereas a document is labeled *bad* if it contains no relevant concepts. Based on observation and experience, for each metric, we chose a low threshold to select bad documents and a high threshold to select good documents. Documents that qualify based on these high/low thresholds are manually validated to make sure they are indeed good/bad documents. The documents that fall between these two thresholds are considered to be of indeterminate quality and hence they are not used for training. We call these documents "grey documents". In our corpus, some grey documents contained terminology present in good documents but were semantically not good while others did not contain the terms found in good documents but were semantically good. We used some of these documents for testing, as shown in Section 5.2.

On the other hand, we believe that adding some grey doc-

uments into the training set may improve the classifier performance. Domain experts should manually validate and select those grey documents to be used in the training set, which is active learning [13, 3, 6], as shown in the right half of Figure 3. We leave that investigation for future work.

Notice that this methodology is domain independent and can be applied to other domains when generating training sets. However, in a different setting, the regular expressions and metrics should be adapted with regard to that specific domain. In Figure 3, we illustrate a feedback loop between the classification model and the domain knowledge representation to further improve the accuracy of the classifier and the understanding of the domain. The solution also makes it possible to create and maintain the domain knowledge representation of the abstract concept by employing such a semi-automated feedback loop. The feedback loop is not in the scope of this paper.

4 Fuzziness Reduction

In text classification systems, features are used to represent the content of text data. In order to obtain a classification model with high accuracy, it is critical that these features are discriminative across different categories. Here we propose an algorithm to disambiguate features in the training data set.

We observed that, in the IT services domain, documents from different categories (e.g., documents relevant to the win strategy concept and those not relevant to the concept) may share a fair amount of terminology, resulting in fuzzy terms present across the categories. Our objective is to automatically select those non-fuzzy terms to form the right feature set.

In our specific binary classification problem in the IT services domain, this is a type of feature selection by which we remove those (fuzzy) terms that do not help in discriminating between good and bad documents because the document frequency of these terms is similar for both classes. Suppose D is the set of documents in the training data (D is the union of good and bad documents), t_k is the kth term in the entire set of terms in D, then we define $W_g(t_k)$ as

$$W_g(t_k) = \frac{|\{d : d \in D \ \& \ t_k \in d \ \& \ d \ is \ a \ good \ document\}|}{|\{d : d \in D \ \& \ t_k \in d\}|}$$

and define $W_b(t_k)$ as

$$W_b(t_k) = \frac{|\{d : d \in D \ \& \ t_k \in d \ \& \ d \ is \ a \ bad \ document\}|}{|\{d : d \in D \ \& \ t_k \in d\}|}$$

Essentially, $W_g(t_k)$ reflects the percentage of good documents in which t_k appears while $W_b(t_k)$ reflects the percentage of bad documents in which t_k appears. We pick

FEATURE SELECTION BASED ON FUZZINESS REDUCTION
/* Input: A set F of features, a set D of documents,
a set of categories C_1, C_2, ..., C_m to which the documents from D belong. */
/* Output: A set of selected features from F */
1) F' $\leftarrow \Phi$
2) for each term t_k in F
3) Count the number of documents containing t_k within each category C_i, where i = 1, 2, ..., m;
4) Compute the weight $W(t_k, C_i)$ of t_k with regard to each category C_i
5) for each pair of C_i and C_j, where i $\neq$ j, compute the difference $|W(t_k, C_i) - W(t_k, C_j)|$
6) if for more than $X\%$ of the category pairs, it is true that $|W(t_k, C_i) - W(t_k, C_j)| \geq T$
7) F' $\leftarrow$ F' $\cup \{t_k\}$;
8) return F';

Figure 4. Algorithm for Fuzziness Reduction

those terms whose W_g and W_b are not too close, i.e., for a term t_k to be selected, the following condition holds:

$$|W_g(t_k) - W_b(t_k)| \geq T$$

Here T is a threshold, which we chose to be 0.2 to maximize the recall in Section 5.

Figure 4 presents the generalized algorithm for fuzziness reduction based on the above idea. Suppose F is the feature set containing all unique terms in the collection, D is the set of documents. C_1, C_2, ..., C_m are the categories to which the documents from D belong. For each term t_k in F, we compute the number of documents that contain the term t_k within each category C_i, where i is between 1 and m, as shown in line 3 in Figure 4. In line 4, the algorithm computes the weight $W(t_k, C_i)$ of t_k with respect to category C_i. This weight is defined as follows:

$$W(t_k, C_i) = \frac{|\{d : d \in C_i \ \& \ t_k \in d\}|}{\sum_{j=1}^{m} |\{d : d \in C_j \ \& \ t_k \in d\}|}$$

By looking at the weights of t_k with regard to different categories, we can decide its commonality across the collection. Therefore, in line 5, we compute the difference of the weights for each pair of categories. If a term is equally popular across different categories, then the term is a fuzzy feature since it does not help discriminate between the categories. This insight is shown in line 6, where, if for more than $X\%$ of the category pairs, it is true that the weight difference of t_k is more than or equal to the threshold T, then t_k is selected and put into the set F' for output. X is a threshold variable. Based on the performance of classification, we can decide the value of X for a term to be selected.

Assume that the number of categories m is much smaller than the number of documents $|D|$ and the number of terms

$|F|$. Then the time complexity of the algorithm in Figure 4 is $O(m \cdot |D| \cdot |F|)$. This is because both line 3 and line 4 are bounded by $O(m \cdot |D|)$. Lines 5 and 6 are bounded by the total number of category pairs, which is still much smaller than $O(m \cdot |D|)$.

Note that the algorithm presented in Figure 4 is a general framework where the weight of a term t_k with regard to a category C_i can be defined in different ways. For example, instead of using document frequency, we can simply use term frequency and define the weight as follows:

$$W'(t_k, C_i) = \frac{f(t_k, C_i)}{\sum_{j=1}^{m} f(t_k, C_j)}$$

Here $f(t_k, C_i)$ is the frequency of term t_k in the category C_i and $\sum_{j=1}^{m} f(t_k, C_j)$ is the sum of term frequency of t_k across all of the categories. However, sometimes, only using term frequency is not enough. For example, a term t_k may appear p times in exactly one document in category C_i while it may appear p times in p documents in category C_j. If we use the function W', then $W'(t_k, C_i)$ is the same as $W'(t_k, C_j)$ so that t_k is identified as a fuzzy feature. To accommodate such subtlety, we also propose to use $tfidf$ weights when computing W:

$$W''(t_k, C_i) = \frac{tfidf(t_k, C_i)}{\sum_{j=1}^{m} tfidf(t_k, C_j)}$$

Here $tfidf(t_k, C_i)$ is the $tfidf$ weight of term t_k in the category C_i and $\sum_{j=1}^{m} tfidf(t_k, C_j)$ is the sum of $tfidf$ weights of t_k in all of the categories.

We can also use a combination of the three defined weighting functions W, W' and W'' to disambiguate the features. For example, we can regard only those features as fuzzy features that are ambiguous based on at least two of the weighting functions. Or, we regard only the features as fuzzy ones that are ambiguous based on all of the three weighting functions. For the experimental results in Section 5, we used the weighting function W, which is based on document frequency. In the future, we will compare the performance of the three weighting functions as well as their combinations.

Our methodology does not require any specific domain knowledge. Therefore, it can be easily applied to other domains. If certain domain specific features should not be considered fuzzy, the algorithm can be slightly modified by taking a list of domain specific terms as input for retaining them in the feature set.

5 Experiments and results

5.1 Preparing Features

5.1.1 Basic Feature Selection

The basic feature set we used is bag-of-words (BOW). The plain-text documents with some meta-data information like document URI and document title are used as input to the SVM classification system. We developed Java code to process each document to generate a feature vector. For stemming, we used the Porter's stemming algorithm [20]. For simple dimension reduction, a stop word list with some common English words, such as pronouns and articles, was used. Words that are less than three characters long were also removed. Additionally, we removed those terms that appeared in only one document in the training set.

5.1.2 Feature Selection using Domain Knowledge

We have incorporated some of our domain knowledge into the selection of features.

Stop word list: We made sure that the stop word list did not contain terms like *win, price, cost,* etc.; these words are important for our domain but may be considered to be noise in other domains.

Filter list: The terms filtered by the fuzziness reduction algorithm are fuzzy terms. To investigate how domain knowledge would possibly affects fuzziness reduction, we selected a subset of those fuzzy terms based on our experience with this domain and call it "filter list". Those terms in the filter list are domain specific and appear frequently enough in both good and bad documents, e.g., *IBM, company, customer, copyright*, etc. We present the performance of using this filter list in Section 5.2.

Feature combinations: The various feature sets that we used in our experiments include: (1) BOW: Bag-of-Words with stemming plus elimination of stop words; (2) BOW-Filter: Bag-of-Words with stemming plus elimination of stop words and elimination of terms in the filter list (those fuzzy terms not in the filter list are still in the feature set); (3) BOW-Filter-FR: Bag-of-Words with stemming plus elimination of stop words and elimination of all of the fuzzy terms including the ones in the filter list. Each of these feature selection techniques improves the performance of the classifier when compared to the use of a basic bag-of-words feature set, as shown in Section 5.2.

5.1.3 Feature Weighting Scheme

Suppose the function $f(t_k, d_j)$ gives the frequency of feature t_k in the document d_j and F is the set of features in the training data. The weight of t_k in the vector of document d_j, $W(t_k, d_j)$, is computed as follows:

$$W(t_k, d_j) = \frac{f(t_k, d_j)}{\sum_{i=1}^{|F|} f(t_i, d_j)}$$

This is normalized term freqency, where the denominator is a normalization technique to prevent bias towards longer documents. In the rest of the paper, we refer to this weighting scheme as our "feature weighting scheme".

We compared the results of the feature weighting scheme mentioned above with the *tfidf* weighting scheme. The *tfidf* weighting method is a widely used conventional weighting scheme used for various classifiers. It emphasizes terms that occur less frequently in the corpus. For a term t_k in document d_j, its term frequency $tf(t_k, d_j)$ is computed the same as $W(t_k, d_j)$ defined above.

Suppose D is the set of documents. Then the inverse document frequency of term t_k is

$$idf(t_k) = \log \frac{|D|}{1 + |\{d \ : \ d \in D \ \& \ t_k \in d\}|}$$

Here the numerator is the total number of documents in the collection and the denominator is the number of documents where t_k appears. To avoid a division-by-zero (in the case where the term is not in the corpus), we add a constant "1" in the denominator. By multiplying term frequency and inverse document frequency, we can get the *tfidf* weight of a term.

We will discuss the performance difference and our observations of using these two different weighting schemes in the Experimental Results (Section 5.2)

5.1.4 Data Set

We collected approximately 600,000 documents from the IT Services engagements that we had access to. In the corpus, only about 1% of the documents contain win strategy concepts of interest to the sales community. Due to the small number of good documents, when using the semi-automated training set algorithm introduced in Section 3, we used relatively high thresholds to choose those obviously good and bad documents. Then we manually validated a subset of those documents. To avoid bias towards bad documents, we selected more or less equal numbers of good and bad documents. We finally complied a total of 573 documents, of which 272 are good and 301 are bad.

5.1.5 Evaluation Metrics

We have used recall, precision, F-measure and overall accuracy to measure the performance. Suppose TP is the number of actual good documents, TN is the number of actual bad documents, FN is the number of actual good documents marked as bad documents and FP is the number of actual bad documents marked as good documents. The four metrics are defined as follows:

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

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

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

$$F - measure = \frac{2 * precision * recall}{precision + recall}$$

5.2 Experimental Results

For our experiments with the SVM algorithm, we used the SVMlight package [1], with the linear kernel using default settings. SVMlight has a parameter to control the trade off for the cost of misclassifying positive and negative documents. For our experiments, the best results were achieved with the cost parameter set to 5000.

In order to measure the robustness of our document filtering classification system, we performed 3-Fold cross validation. Besides the data set mentioned in Section 5.1.4, we chose some additional documents from the set of grey documents. Totally, we added 51 grey documents into the test set, of which 28 were good and the rest were bad.

We computed the precision, recall, F-measure and accuracy for each of the three folds using different feature sets, and present the results in Tables 1, 2, 3, and 4, respectively. This set of results are based on the feature weighting scheme.

3-Fold	BOW %	BOW-Filter %	BOW-Filter-FR %
Fold1	90.4	89.47	87.59
Fold2	87.1	88.15	87.59
Fold3	86.82	86.76	85.31
Average	88.1	88.1	86.83

Table 1. Precision in percentage for each of the 3-Folds using different feature combinations

Table 2 shows a low average recall if only BOW is used as the feature set, indicating that some documents that are good are being misclassified as bad and they will be dropped from any further content processing. However, if we use fuzziness reduction with filter list only (BOW-Filter), then the recall goes up by 7% with the same precision; if we use fuzziness reduction with the complete set of

3-Fold	BOW %	BOW-Filter %	BOW-Filter-FR %
Fold1	90.4	97.54	98.36
Fold2	89.26	98.35	99.17
Fold3	91.8	96.72	100.00
Average	90.5	97.5	99.2

Table 2. Recall in percentage for each of the 3-Folds using different feature combinations

3-Fold	BOW %	BOW-Filter %	BOW-Filter-FR %
Fold1	90.4	93.33	92.66
Fold2	88.16	92.97	93.02
Fold3	89.24	91.47	92.08
Average	89.3	92.6	92.6

Table 3. F-measure in percentage for each of the 3-Folds using different feature combinations

fuzzy terms (BOW-Filter-FR), the recall goes up by almost 9%, but precision is reduced only by slightly more than 1%, which is acceptable because the increase in recall is more significant than the decrease in the precision for an SVM classifier being trained for document selection. Figure 5 shows the gain of recall versus precision of using fuzziness reduction with the feature weighting scheme.

It is interesting to note that BOW-Filter-FR achieves almost 100% recall. Compared to BOW-Filter-FR, BOW-Filter has a lower average recall, but slightly higher precision. Therefore, they have more or less the same F-measure and accuracy performance (see Tables 3 and 4). The observation is that if precision is not a big concern, it is better to use fuzziness reduction with the complete set of fuzzy terms, i.e., the BOW-Filter-FR feature set. Especially in our case of document selection, we need to achieve a re-

3-Fold	BOW %	BOW-Filter %	BOW-Filter-FR %
Fold1	90.24	93	92.18
Fold2	88.07	92.59	92.59
Fold3	88.89	90.95	91.36
Average	89.1	92.2	92

Table 4. Accuracy in percentage for each of the 3-Folds using different feature combinations

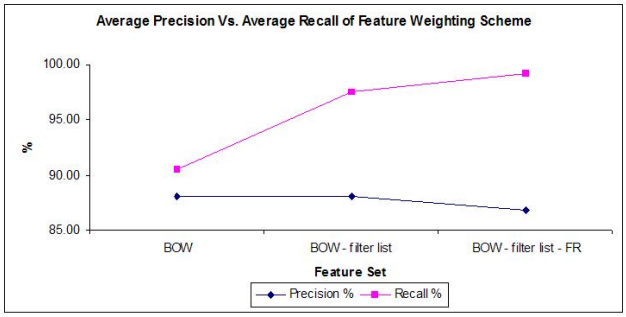

Figure 5. Average Precision vs. Average Recall of Feature Weighting Scheme

call as high as possible. On the other hand, adding certain domain knowledge into feature selection helps identify domain-specific fuzzy terms in the feature set and helps improve the precision, as indicated by the performance of BOW-Filter.

5.2.1 Results of Using *tfidf* Weighting Scheme

We also repeated the same set of experiments as discussed above using the *tfidf* scheme to compute feature weights. Table 5 shows the average precision and recall measures for the *tfidf* weighting scheme. Figure 6 shows the gain of recall versus precision of using *tfidf* together with fuzziness reduction.

Metrics	BOW %	BOW-Filter %	BOW-Filter-FR %
tfidf Precision	89.39	89.5	89.74
tfidf Recall	87.66	88.76	90.96

Table 5. Performance of *tfidf* Weighting Scheme with Fuzziness Reduction

In Figure 7, we compare the recall (left) and precision (right) performance with respect to various feature sets between *tfidf* and the feature weighting scheme. As shown in the comparison, the precision measure using *tfidf* was consistently higher by approximately 2% for all of the feature sets. On the other hand, the recall of using *tfidf* was significantly lower than using the other weighting scheme. Both weighting schemes have the best recall for the BOW-Filter-FR feature set.

We believe that the feature weighting scheme boosted those "good" features such that it brought some of the template documents and general discussion of the win strategy concept into the set of good documents, which were actually

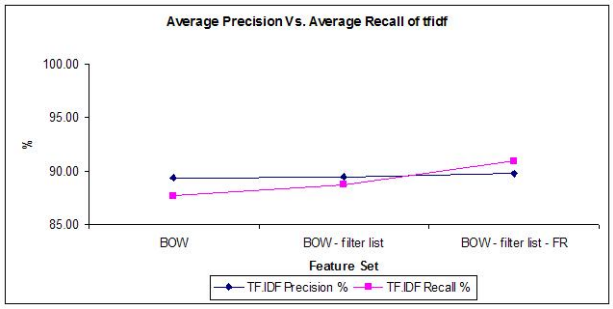

Figure 6. Average Precision vs. Average Recall of *tfidf*

false positives. Oppositely, *tfidf* missed some good documents and put them into the set of bad documents, which became false negatives.

5.2.2 Discussion

The experiments and results presented in this section show that our system is effective, generally applicable, and can be the basis of a robust solution for document selection using SVM, making it possible to automatically classify good and bad documents. In our experiments, we have used the precision, recall, F-measure, and accuracy to measure the effectiveness of the various feature sets.

For our document selection, we are interested in a very high recall, because we do not want to drop a good document. On the other hand, it is acceptable if the classifier misclassifies some bad documents as false positives, because during the extraction phase we can always reject these documents if they do not have any relevant pieces of information suitable for extraction.

5.3 Baseline

We started document filtering with the heuristic rule based approach, which is thus the natural baseline for our SVM classifier. The major idea of the heuristic rule based approach is to apply a set of regular expressions to the metadata of each document, including document title and URL. These regular expressions are human crafted and encode our knowledge of the IT Services domain. For example, we know from experience that certain kinds of documents, such as executive summaries, are good candidates for win strategy extraction. If a document falls into that category since its title satisfies the corresponding regular expressions, that document would be marked as a good document. On the other hand, certain documents types typically do not include any win strategy information, e.g., those whose titles indicate that they are templates. We can identify those types of documents by checking their URL and mark them as bad documents.

Table 6 presents the performance of the heuristic rule based approach using the same test set used in the SVM experiments (see Section 5.2). As indicated by the average recall, certain amount of good documents have passed the test, but this approach introduces a lot of noise into the results, which makes the task of identifying right passages for extraction even more challenging. In addition, our goal is to achieve 100% recall if possible so that we do not miss any good documents in the extraction step. From that perspective, the SVM approach has performed much better than the heuristic rule based approach.

3-Fold	Precision %	Recall %	F-Measure %	Accuracy %
Fold1	69.4	82	75.2	72.8
Fold2	68.5	82.6	74.9	72.4
Fold3	64.1	80.3	71.3	67.5
Average	67.3	81.6	73.8	70.9

Table 6. The Performance of Heuristic Rule Based Document Filtering

5.4 Deployment

Recently, our EIL search platform has been deployed in a production environment with 1 million documents from about 3500 IT Services engagements. In addition, the extracted semantic content has been fed into a knowledge sharing repository for sales practitioners. A user study is currently underway to keep track of user experience and usage patterns.

6 Related work

Feature selection has been studied widely[14, 19]. Among various well-known techniques are document frequency (DF), term frequency inverse document frequency (TFIDF), term frequency variance (TfV), latent semantic indexing (LSI), random projection (RP), and independent component analysis (ICA) [12]. DF focuses on choosing top K features based on document frequency, that is, the number of documents that contain these features [27]. Based on TFIDF top K features are selected with high term frequency but low document frequency across a collection. In TfV [8], the basic idea is to rank a feature based on variance of its term frequency. All of the three feature selection methods mentioned here select the top K features from the entire collection based on some threshold without considering category specific information. Another widely studied

Figure 7. Recall (*left*) and Precision (*right*) Comparison Between *tfidf* and Feature Weighting Scheme)

set of techniques is based on feature transformation that is to transform the vector space representation of the document collection into lower dimensional space, which is linear combination of the original dimension. LSI, RP, and ICA are all feature transformation based techniques, which are complex to understand and need longer execution time. Additionally, they are not designed to discriminate features across different categories. On the other hand, quite a few efforts have been put into discovering new textual features based on term relationships and on external resources other than the training data, such as using a thesaurus.

Many other currently known solutions for feature selection are primarily based on manual effort from extensive domain expertise (e.g., [25, 18, 21]). Manual feature selection poses three major drawbacks. First, it is highly time consuming, since an expert has to understand and analyze a large volume of data. Second, manual effort makes the feature selection process slow and limited to expert knowledge. Finally, manual extraction does not necessarily lead to an unambiguous feature set, which is needed for optimal performance.

For training data generation, various approaches have been proposed in the literature. Geyer[13] leveraged semi-automated active learning techniques for software reverse engineering, using domain specific text like user manuals on the software system to construct an ontology and turn the ontology into a concept keywords document which was used as a text filter to label the class names. Then the machine classified examples were presented to the user for confirmation. Though the overall methodology at the high level is similar to what we present in this paper, the difference is the novelty by which documents are classified in our work (i.e., leveraging multi-level pattern matching of regular expressions with total coverage of the concept related semantic annotations).

Bilenko and Mooney[3] proposed approaches based on static active learning for duplicates detection using *tfidf* vector space distance, where non-duplicates selected from the detected duplicates would be "near-miss" negative examples. This paper also suggests that approaches like ours in choosing obviously negative examples are good when the training set is not very large.

The paper from Collins et.al [6] considers active learning for automated dataset collection in computer vision research. This is in a setting where images are collected for a visual category by automatic internet search and relevant images need to be separated from noise. In every iteration of active learning, the system presents to the user the subset of images for which the predicted class is least certain. In our case, due to the high intra-class variation, we restricted the training set to be comprised of highly likely positives, near-miss negatives and highly likely negatives along with other manually selected examples.

Another set of related work are SVM related applications, but none of them is in the IT services domain. For example, [22] presents a system where SVM was used to classifiy call transcripts into sections such as "Greeting Section" and "Question Section". The author leveraged additional features besides words in transcripts, for example, speaker's identity and call section type of the previous section. The average accuracy of the system is 87.2%.

In [9], an SVM was trained using the bag-of-words feature selection method, over MEDLINE abstracts, to distinguish abstracts containing information on protein-protein interactions. A small evaluation with 100 abstracts found a precision of 96% with a recall of 84%.

Recently, the SVM technique was used in a study to assist in the screening of PubMed abstracts for translation of human genome discoveries into health applications [28]. They proposed a two-way z-score method to obtain statistically significant weighted keyword features for classification. This method achieves 97.5% recall and 31.9% precision in performance testing.

7 Conclusions and Future Work

This paper presents a system to apply machine learning techniques – a SVM classifier – in selecting the right can-

didate documents for extracting concepts in the IT services domain. To the best of our knowledge, this is the first time that a SVM classifier is used to classify services engagement documents.

One of the difficulties in extracting concepts in this domain is the fuzziness of both documents and concepts due to the following reasons: (1) The concepts we are targeting are normally business concepts (e.g., win strategy). The definition of these concepts is not clear in most cases. (2) The documents in this domain are normally fuzzy. For example, good and bad examples share many terms. Some documents are bad examples semantically, but syntactially they are good, i.e., they contain many good features. We show how to use domain knowledge in the feature selection phase to improve the performance of the SVM classifier. Specically, we have proposed a feature selection technique to reduce the fuzziness in the feature vector. The experimental results show that these techniques have significantly improved the performance of the classifier, especially the recall.

Another major contribution of this paper is the development of a methodology to semi-automatically generate a training set by using domain knowledge. It is generally a time consuming and tedious task to manually mark good and bad documents. We have successfully adapted annotator techniques to mark up words and phases that are important to the IT services domain. Based on that, we defined a set of metrics to select very good/bad examples. During the manual validation phase, the annotations can guide domain experts to navigate through documents efficently and effectively.

In the future, we plan to continue exploring the use of domain knowledge to improve the performance of our SVM classifier, in particular, its precision. We would also like to compare our SVM classifier with other classifiers in which other types of features, such as noun phrases, are needed. Finally, we plan on closing the loop for win strategy concept extraction by using SVM not only to filter the appropriate candidate documents but to extract the most relevant sentences and passages from these documents.

References

[1] SVM-Light Support Vector Machine. http://svmlight.joachims.org/.

[2] Science of selling. *IDC Insights*, 2007.

[3] M. Bilenko and R. J. Mooney. On evaluation and training-set construction for duplicate detection. In *Proceedings of the KDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, pages 7–12, Washington, DC, 2003.

[4] A. Boyd and N. Le. Sales effectiveness: Getting sales back to selling. *Aberdeen Research Report*, 2007.

[5] F. Ciravegna. Challenges in information extraction from text for knowledge management. *IEEE Intelligent Systems and Their Applications*, Nov. 2001.

[6] B. Collins, J. Deng, K. Li, and F.-F. Li. Towards scalable dataset construction: An active learning approach. In *Proceedings of ECCV*, pages 86–98, Berlin, Heidelberg, 2008.

[7] Y. Deng, M. Devarakonda, N. Rajamani, and W. Zadrozny. Improving information access for a community of practice using business process as context. In *Proceedings of ICDE*, Cancun, Mexico, April 7-12, 2008.

[8] I. Dhillon, J. Kogan, and M. Nicholas. Feature selection and document clustering. *M.W. Berry, editor, A Comprehensive Survey of Text mining*, 2003.

[9] I. Donaldson, J. Martin, B. de Bruijn, and C. Wolting. PreBIND and textomy - mining the biomedical literature for protein-protein interactions using a support vector machine. *BMC Bioinformatics*, 4:11–23, 2003.

[10] S. Feldman, J. Duhl, J. R. Marobella, and A. Crawford. The hidden costs of information work. *IDC White Paper*, Mar. 2005.

[11] D. Ferrucci and A. Lally. Building an Example Application with the Unstructured Information Management Architecture. *IBM Systems Journal*, 43(3):455–475, 2004.

[12] I. Fodor. A survey of dimension reduction techniques. *Technical report UCRL-ID-148494, LLNL*, 2002.

[13] J. M. Geyer. Using ontology creation, text filtering, and active learning to generate training sets. *Thesis proposal, Miami Distributed Enterprise Systems and Software Engineering Laboratory*, 2009.

[14] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

[15] T. Jenssen, A. Laegreid, J. Komorowski, and E. Hovig. A literature network of human genes for high-throughput analysis of gene expression. In *Pacific Symposia in Biocomputing*, 2002.

[16] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML*, pages 137–142, 1998.

[17] T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of ICML*, pages 200–209, 1999.

[18] W. Li and A. McCallum. Semi-supervised sequence modeling with syntactic topic models. In *Proceedings of AAAI*, pages 813–818. AAAI Press, 2005.

[19] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. on Knowl. and Data Eng.*, 17(4):491–502, 2005.

[20] MF Porter. An algorithm for suffix stripping. *Program*, 14:130–137, 1980.

[21] S. Miller, J. Guinness, and A. Zamanian. Name tagging with word clusters and discriminative training. In *HLT-NAACL*, pages 337–342, 2004.

[22] Y. Park. Automatic call section segmentation for contact-center calls. In *Proceedings of CIKM*, pages 117–126, 2007.

[23] N. Rajamani, M. Devarakonda, Y. Deng, W. Zadrozny, and J. Pathak. Business-activity driven search: Addressing the information needs of services professionals. In *International Conference on Services Computing*, Utah, USA, 2007.

[24] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[25] X. Wei, A. Sailer, R. Mahindru, and G. Kar. Automatic structuring of it problem ticket data for enhanced problem resolution. In *Integrated Network Management*, pages 852–855, 2007.

[26] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of SIGIR*, Berkeley, California, USA, 1999.

[27] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of ICML*, pages 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[28] W. Yu, M. Clyne, S. M. Dolan, A. Yesupriya, A. Wulf, T. Liu, M. J. Khoury, and M. Gwinn. Gapscreener: An automatic tool for screening human genetic association literature in pubmed using the support vector machine technique. *BMC Bioinformatics*, 9, 2008.