Text Categorization in Indian Languages using Machine Learning Approaches

K Raghuveer and Kavi Narayana Murthy
Department of Computer and Information Sciences,
University of Hyderabad,
Hyderabad, 500 046,
email: raghuveer_ai@yahoo.co.in, knmuh@yahoo.com

Abstract

In this paper we present our work on automatic text categorization in Indian languages. Here we use purely corpus based machine learning techniques. The methods we present are completely language independent - no language specific knowledge is used. We describe our experiments on ten of the major Indian languages including Assamese, Bengali (Bangla), Gujarati, Hindi, Kannada, Malayalam, Oriya, Punjabi, Tamil and Telugu. We have conducted our experiments on the DoE-CIIL corpora. We have also worked on the newspaper corpus forming part of the LERC-UoH Telugu corpus developed by us. We have used several machine learning techniques including naive Bayes classifier, k-Nearest-Neighbor classifier and SVMs. We have used one-vs.-all SVMs for multi-classification with 3-fold Cross Validation in all cases. We see that SVMs out-perform other classifiers. We describe our experiments with soft-margin linear SVMs as well as kernel based SVMs using polynomial and Radial Basis Function kernels. Kernel based SVMs have not performed significantly better than linear SVMs.

There is not much work done in text categorization in Indian languages. Text categorization in Indian languages is challenging as Indian languages are very rich in morphology, giving rise to a very large number of word forms and hence very large feature spaces. We show how Mutual Information between features and categories can be used to achieve substantial reduction in the dimensionality of the feature space without reducing the performance. In fact many terms actually act as noise and we show that pruning such terms from the feature space actually enhances the performance.

The paper is written in tutorial style and adequate background material is included on text categorization as also on the machine learning techniques used, for the benefit of readers who may not be familiar with these.

Keywords : Text Categorization, SVM, naive Bayes, kNN, Mutual Information, Indian Languages

1 Introduction

Over the recent past, there has been an explosion in the availability of electronic information. As the availability of information increases, the inability of people to assimilate and profitably utilize such large amounts of information becomes more and more evident. The most successful paradigm for organizing this mass of information, making it comprehensible to people, is perhaps by categorizing the different documents according to their subject matter or topic. Automatic text categorization has many applications including indexing for Information Retrieval Systems and Search Engines, Document Organization, Text Filtering (emails, for example), News Aggregation and Organization, and Word Sense Disambiguation.

1.1 Text Categorization Defined

Automatic text categorization can be defined as assigning pre-defined category labels to new documents based on the likelihood suggested by a training set of labeled documents. It is the task of assigning a value to each pair $(d_j, c_i) \in DXC$ where D is a domain of documents and C is a set of predefined categories. Performance can be measured in terms of Precision and Recall, or using a combined measure such as the F-measure.

There are several variations to this basic theme:

– Constraints may be imposed on the number of categories that may be assigned to each document - exactly k, at least k, at most k, and so on. In the *single label* case, k = 1 and a single category is to be assigned to each document. If k is more than 1, we have *multi-label* categorization.

– Text categorization problem can be reduced to a set of binary classification problems, one for each category - where each document is categorized into either $c_i or \overline{c_i}$.

– In *Hard categorization*, the classifier is required to firmly assign categories to documents (or the other way around) whereas in *Ranking Categorization*, the system ranks the various possible assignments and the final decision about class assignments is left to the user. This leads us to the possibility of semi-automatic or interactive classifiers where human users take the final decisions to ensure highest levels of accuracy.

– In *Document Pivoted Categorization* a given document is to be assigned category label(s) whereas in a *Category Pivoted Categorization*, all documents that belong to a given category must be identified. This distinction is more pragmatic than conceptual. Thus if all the documents are not available to start with, document pivoted categorization may be more appropriate while category pivoted categorization may be the preferred choice if new categories get added and already classified documents need to be re-classified.

– If only unlabeled training data is available we may have to use unsupervised learning techniques to perform *Text Clustering* instead of classification into known classes.

In this work, single label binary categorization on labeled training data is carried out on Indian languages.

1.2 Representing Text

– *Word Level:* Words form a natural representation for texts. In most cases simply tokens separated by white spaces are treated as words. Compounds and phrases are not analyzed. Polysemy and homonyms are also ignored. Texts are represented simply in terms of surface forms of words. In addition syntactic structure is ignored and it is assumed that order of the words is irrelevant. This representation is known as *bag of words* representation. Despite its obvious limitations, the bag of words representation is often found to be quite effective in information retrieval and text classification tasks.

– *Sub-word Level:* n-grams are the most popular at the sub word level. Here, instead of using words, n-character strings are used as the basic building blocks. For example, the 3-gram (or trigram) representation of the word ”lead” is ”␣le”, ”lea”, ”ead”, ”ed␣”. The whole document is represented as a bag of these basic building blocks. n-gram representation is basically a model of the similarity between words. While *computer* and *computing* are different surface words they share a large percentage of trigrams. However, trigram similarity can be misleading too as in *computer* and *commuter*

– *Multi Word Level:* Representations at the multi word level generally use indexing terms that incorporate syntactic structure. This approach is commonly known as Syntactic Phrase Indexing. Another approach for generating multi word indexing terms is based on statistical methods. Here co-occurrence patterns are analyzed.

– *Semantic Level:* Clearly, text classifiers can only work optimally if one can capture the semantics of documents sufficiently. Unfortunately it is not easy to automatically extract semantics of free text. Latent Semantic Indexing is one method which aims to automatically generate semantic categories based on bag-of-words approach [1].

Whatever be the choice, the units of text representation are called ’terms’. In text categorization, terms are the features.

In this paper we have used word level representation. A document is treated as a bag of D words and is viewed as a point in D-Dimensional space, leading to the well known Vector Space model [2].

1.3 Feature Selection and Dimensionality Reduction

Feature dimensions in text categorization tend to remain very large, often running into tens of thousands. In Indian languages, the numbers will be even higher, given the richness of morphology. *The curse of dimensionality* is that the number of training data samples required grows exponentially with the number of features. Choice of the right subset out of the potential feature set is therefore a major concern. A variety of dimensionality reduction techniques are used in pattern recognition but applying them for text categorization requires care. Below we list some of the commonly used techniques.

– *Stop Words* are considered noise and removed. Stop words may be defined in several ways. Syntactically, they are the function words (determiners, prepositions, conjunctions etc.). They usually occur very frequently and tend to be small words. A stop word dictionary may be maintained. Stop word lists have not yet been well established and widely available in Indian languages.

– *Morphology and Stemming:* Inflected words mean essentially the same as their corresponding root forms and thus reducing full words to their root forms reduces the variability. Complexity of morphological analysis may vary greatly and a stemming algorithm may be used instead of full morphological analysis. Table look-up, iterative affix removal and n-grams based stemmers have been proposed. Morphology of Indian languages is very complex. Dravidian languages, in particular, are among the most complex in the whole world. Analysis of a 40 Million word corpus of Telugu language for example, showed that there were more than 3.3 Million different word forms and more forms are to be expected as the growth-rate curves have not shown any signs of saturation [3]. Wide coverage, high performance morphological analyzers and stemmers are not yet available for most Indian languages.

– *Phrases:* Phrases are typically contiguous words that behave as atomic units of syntax and semantics. Syntactic phrases or word-groups can be efficiently identified through a Finite State grammar [4]. A statistical phrase is defined by constraints upon the frequency of occurrence of the phrase, upon the co-occurrence of its components, and/or upon the proximity of its components in the texts. Semantic phrases or idioms are usually ignored in text categorization. A dictionary of phrases may be maintained. In some systems, function words within phrases are omitted and the remaining content words are treated as unordered - a kind of normalization. A variety of techniques exist for handling proper names and other named entities. Here again, developments in Indian languages are slow.

– *Collocations:* Terms that co-occur frequently in various categories may have significance to classification into those categories.

– *Controlled Vocabulary:* In some situations, it may be possible to use a predefined set of index terms instead of attempting to obtain the index terms from the collections of documents given. Terms from a subject thesaurus, subject headings, classification codes etc. may be used.

– *Mutual Information:* Mutual information is one of the most common measures of relevance. It measures the reduction in entropy obtained by considering two random variables together. High mutual information indicates closer relationship. We can measure the mutual information between potential features and categories and only those features with high mutual information may be retained. Mutual Information is defined as follows:

$$\begin{aligned} I(Y,W) &= H(Y) - H(Y/W) \\ &= \sum_{y \in \{-1,+1\}} \sum_{w \in \{0,1\}} Pr(y,w) \log \frac{Pr(y,w)}{Pr(y) \cdot Pr(w)} \end{aligned}$$

Here random variable Y indicates the class label assigned to a document. The random variable W indicates whether the particular word occurs in the given document are not. Entropy H(Y) is a measure of the uncertainty in the random variable Y. I(Y,W) describes the information that word W contributes to the class Y. Only the terms with high mutual information are selected as features.

Since morphological analyzers and stemmers of adequate performance are not yet available for many Indian languages, in this work we have used surface forms of words as features. We use mutual information for dimensionality reduction. We find mutual information for every word to every category. For every category we sort all the words based on the mutual information. We select words whose mutual information is above a threshold. The thresholds are carefully chosen by experimentation. It may be reiterated that this technique is language independent and no linguistic knowledge is required. Also, the distribution of features across categories may not be uniform and this aspect has been taken into consideration while selecting features. We select features that help to characterize different categories of documents and help distinguish one class from the other.

1.4 Feature Weighting

Numerical weights are to be computed for the index terms before machine learning techniques can be applied. Here are some of the basic techniques for term weighting:

- *Term Attributes:* Attributes of the terms such as their syntactic categories can be used to weight the terms.
- *Text attributes:* The number of terms in a text, the length of the text etc. can be used.
- *Relation between the term and the text:* Relative frequency of the term in the text, location of the term in the text, relationship with other terms in the text etc. can be used.
- *Relation to corpus:* Relation between the term and the document corpus or some other reference corpus can also be used.
- *Expert Knowledge:* Expert knowledge is a potential source but is rarely used.

– *Term Frequency:* Words that occur more frequently in various categories are believed to be more significant in classification and are thus given higher weightage. Since the occurrence of a rare term in a short text is more significant than its occurrence in a long text, log of the term frequency is used to reduce the importance of raw term frequencies in those collections that have a wide range of text lengths. Anaphoric references and synonyms reduce the true term frequency. In morphologically rich languages, poor stemming also adds to the same effect.

– *Inverse Document Frequency* Terms that occur in (almost) all documents are useless for classification. Terms that occur in smaller number of documents are given higher weightage. *Inverse Category Frequency* would be more appropriate than just inverse document frequency since the distribution of documents into categories may be skewed. A log is again taken often to weigh down this aspect so that it does not become over dominating.

– *Product of tf and idf:* Term frequency and Inverse Document Frequency are inter-related. Terms that frequently in a particular class but not very frequently in other classes are the most significant. Hence a product of tf and idf is often used. The tf-idf feature weighting scheme is one of the most widely used class of feature weighting methods for text categorization.

– *Length Normalization:* Long and verbose texts usually use the same terms repeatedly. As a result, the term frequency factors are large for long texts and small for short ones, obscuring the real term importance. Term frequencies can be normalized for length of texts by dividing them with the frequency of the most frequently occurring term in the text. Another method of length normalization is the cosine normalization where each term weight is divided by a factor representing the Euclidean vector length.

These various ideas can be combined to produce a final term weight(tf-idf):

$$\frac{tf_i * log(\frac{N}{n_i})}{\sqrt{\sum_{j=1}^{n}(tf_j * log(\frac{N}{n_j}))^2}}$$

where tf_i is the term frequency for term i, N is the number of documents in the collection, n_i is the number of documents in the collection that include the index term i, and $j = 1..n$ is the number of distinct index terms in the text.

The distribution of terms across categories is also very important. In this work we use normalized tf-idf products computed category-wise[5] as given below. First the tf values are computed for each term in each document and normalized by dividing by the frequency of the most frequent term in the document. The feature value for each term w, for category C_i is the computed as:

$$tfidf(w|C_i) = \frac{tf_i(w) * log(\frac{N}{n(w)})}{\sqrt{\sum_{j=1}^{n}(tf_j(w) * log(\frac{N}{n(w)}))^2}}$$

where $tf_i(w)$ is the term frequency for term w in category i, N is the number of documents in the collection, $n(w)$ is the number of documents in all the categories that include the index term w, and $j = 1..n$ are the categories.

2 Complexity of Indian Languages

India is a country of one Billion people, nearly one sixth of the whole world. India is also an ancient civilization, dating back to many thousands of years. The Indian subcontinent consists of a number of separate linguistic communities each of which share a common language and culture. Some Indian languages have a long literary history. Sanskrit literature is more than 5,000 years old. Contrary to common belief, there are more than 150 different languages spoken in India today. These are not dialects - dialects add up to a much larger number. Many of the languages have not yet been studied in any great detail. Of these, 22 major languages have been given constitutional recognition (apart from English). These major languages include the official languages of the federal states of India and are among the most widely spoken languages of the world. These languages span four different language families - the Indo-Aryan, the Dravidian, the Tibeto-Burman and the Austro-Asiatic families[6].

2.1 Richness in Morphology

Modern Indian languages all have close ties with Sanskrit and are characterized by a rich system of inflectional morphology and a productive system of derivation, saMdhi (conflation of full words) and compounding. This means that the number of surface words will be very large and so will be the raw feature space, leading to data sparsity.

Dravidian morphology is in particular more complex. Dravidian languages such as Telugu and Kannada are morphologically among the most complex languages in the world, comparable only to languages like Finnish and Turkish. Below we only give a glimpse of the nature and complexity of Telugu morphology. See [3] for more details. The main reason for richness in morphology of Telugu (and other Dravidian languages) is, a significant part of grammar that is handled by syntax in English (and other similar languages) is handled within morphology. Phrases including several words in English would be mapped on to a single word in Telugu. Thus 'vaccaaDu' ((he) came), 'vastaaDaa' (will (he) come?), vaste (if (he/she/it/they/I/we/you) come), 'ragalagutaaDu' ((he) will be able to come), 'raaleekapooyaaDu' ((he) was unable to come), 'vaccinavaaDu' (the person (3P,sl) who came), 'raaDanukonnaavaa' (do you think he will not come?) are all single words in Telugu, written and spoken as atomic units without spaces or pauses. Verbs may include aspectual auxiliaries apart from tense and agreement. There are several types of non-finite forms too. A single verbal root can lead to the formation of hundreds of thousands of word forms. Nouns are inflected for number and case. Derivation being very productive, even more forms become possible when we consider full word forms. Thus 'vaccinavaaDiki' (to the person (3P, sl) who came) is a noun in singular, dative case derived from the verb root 'vacc' (to come). External saMdhi (that is, conflation between two or more complete word forms) and compounding add to the numbers. Naturally we will see very large number of types and the type-token ratio should be expected to be very high too. These are not simple concatenations or juxtapositions of complete words as is the case in some languages of the world. These words are made up of several morphemes conjoined through complex morpho-phonemic processes. The LERC-UoH Telugu text corpus developed by us here, adds up to nearly 40 Million words in total size and includes as many as 330,0000 different words! And our analyses show that this list is far from complete and many more forms should be expected as we develop larger corpora [3].

One way to handle this large and sparse feature space is to employ morphological analyzers or stemmers to reduce surface words to their root or stem forms. Developing high performance morphological analysers and stemmers has, however, remained a difficult challenge till date for many Indian languages. In this paper, we instead use purely corpus based statistical techniques to identify the most promising features and prune the rest. We show that mutual information between terms and categories is a simple yet very effective dimensionality reduction technique.

3 Techniques for Text Categorization

Before the 1990s, the predominant approach to text classification was the knowledge based approach. With the increasing availability of large scale data in electronic form and advances in machine learning and statistical inference, there has been a clear shift in the approach towards automatic learning from large scale data over the last decade or so. In the Machine Learning approach, a general inductive process (also called the learner) automatically builds a classifier for a category c_i by observing the characteristics of a set of training documents already classified under $c_i or \overline{c_i}$. The inductive process gleans from these labeled training data the characteristics that a new unseen document should have in order to be classified under c_i. The classification problem is thus an activity of supervised learning.

An increasing number of learning approaches have been applied, including Regression Models [7, 8], Nearest Neighbor Classification [9, 10, 11, 12, 13], Bayesian Probabilistic Approaches [14, 15, 16, 17, 18, 19, 20], Decision Trees [7, 15, 16, 18, 21], Inductive Rule Learning [22, 23, 24, 25], Neural Networks [26, 27], On-line Learning [24, 28], and Support Vector Machines [18][29]. Yang and Liu [30] have made a systematic comparative study of several of these approaches and concluded that all methods perform comparably when the distribution of documents across categories is more or less uniform.

3.1 Bayesian Learning Methods

Bayesian Learning is a probabilistic approach to inference based on the assumption that the quantities of interest are governed by probability distributions and the optimal decision can be made by reasoning about these probabilities together with observed data.

In Bayesian Learning methods a maximum a posteriori (MAP) probability is computed using the Bayes Theorem. The basic idea is to use the joint probabilities of terms and categories to estimate the probabilities of categories given a document.

In some cases, the prior probabilities of all the hypotheses are assumed to be uniform and hence bracketed out. This assumption of uniform priors is questionable and has led to criticism of the Bayesian approaches.

Bayesian method requires the estimation of joint probabilities of all the features for each category. In order to simplify this, independence is often assumed. That is, the conditional probability of a feature given a category is assumed to be independent of the conditional probabilities of other features given that category. A Bayesian classifier that makes this independence assumption is termed a naive Bayes Classifier. The Independence assumption is rarely valid. Yet the method works and is used in practice [15, 16, 17, 20, 19].

To categorize a test document d_j as belonging to a category C_i, the maximum likelihood is estimated over all categories:

$$P(d_j|C_i) = \sum_{w \in d_j} log(P(w|C_i))$$

The prior probabilities of each category - Prior(C_i) - are evaluated as the ratios of the number of documents in category C_i to the number of documents in the total collection.

Finally, the posterior probabilities of each category are calculated by adding the log likelihoods to the log priors.

$$P(C_i|d_j) = log(P(d_j|C_i)) + log(Prior(C_i))$$

A test document is assigned the category with the maximum posterior probability. To minimize misclassification errors due to narrow differences, a threshold value can be used to include a reject option. Performance can then be measured in terms of Precision, and Recall. In order to capture the Precision-Recall trade-off in a single quantity, a combined measure such as the F-measure can be used.

3.2 k-Nearest Neighbor Classifier

The k-nearest neighbor (kNN) algorithm is an example of instance based learning [31]. Given a test document, the kNN algorithm finds the k nearest neighbors among the training documents and uses the categories of the k nearest neighbors to assign a category to the test document. In the simplest case, simply the majority class is assigned. In distance weighted kNN, the similarity score of each neighbor document to the test document is used as the weight of the categories of the neighbor document. If several of the k nearest neighbors share a category, then the per-neighbor weights of that category are added together and the resulting weighted sum is used as the likelihood score of that category with respect to the test document. The scores of the candidate categories are sorted and a ranked list is obtained for the test document. Thresholding can then be applied to obtain binary category assignments. Category specific thresholds can be learned from a set of validating documents by looking for the threshold that maximizes a performance measure such as the F-measure. kNN is a simple and well explored technique that has been found to perform well for many problems including text categorization [9, 10, 32]. For the case of binary classification, we have:

$$h_{knn}(\overrightarrow{x}) = sign\left(\frac{\sum_{i \in knn(\overrightarrow{x})} y_i cos(\overrightarrow{x}, \overrightarrow{x_i})}{\sum_{i \in knn(\overrightarrow{x})} cos(\overrightarrow{x}, \overrightarrow{x_i})}\right)$$

Here x is input vector, y_i is class label and $knn(\overrightarrow{x})$ denotes the indices of the k documents which have the highest cosine with the document to classify $\overrightarrow{x}$.

3.3 Support Vector Machine

Support Vector Machine (SVM), a binary classifier, was developed by Vapnik based on the Structural Risk Minimization (SRM) principle from statistical learning theory [33] [34] [29]. The idea is to find a hypothesis h for which we can guarantee lowest error $Err(h)$. True error is the probability that h will make an error on unseen data. The following upper bound connects the true error of a hypothesis h with the $Err_{train}(h)$ of h on the training data and the complexity of h:

$$Err(h) \leq Err_{train}(h) + O\left(\frac{d \ln(\frac{n}{d}) - \ln(\eta)}{n}\right)$$

The bound holds with a probability of at least $1-\eta$. Here n is the number of training examples, d denotes the VC-dimension, which is a property of a hypothesis space h, which indicates its expressiveness. A simple hypothesis space (low VC dimension) will not contain a good approximation function and will lead to a high training error. On the other hand too rich a hypothesis space will lead to small training error, but the second term in the above equation will be large, which will cause increase in the true error. This indicates over-fitting. So it is crucial to pick up a hypothesis space with the right complexity. In SRM this is done by defining a nested structure of hypotheses space H_i so that the respective VC-dimension d_i increases from the smallest hypothesis space to the largest.

$$H_1 \subset H_2 \subset H_3 \subset ... \subset H_i \subset ... \qquad and \qquad \forall : d_i \leq d_{i+1}$$

The goal is to find the index i^* for which error $Err(h)$ will be minimum. SVMs find such a hypothesis space by maximizing the margin. One remarkable property of SVMs is that their ability to learn is independent of the dimensionality of the feature space. SVMs measure the complexity of hypothesis space based on margin rather than dimensionality of the feature space.

3.3.1 Linear Hard-Margin SVM

Let us assume that the training data can be separated by at least one hyperplane h_i. This means that there is weight vector w and threshold b so that all positive training examples are on one side and all negative training examples are on the other side of hyperplane. From all possible separating hyperplanes, SVM will choose the one with maximum margin. Finding a hyperplane with maximum margin can be translated into the following optimization problem:

$$\begin{aligned} &minimize_{w,b} && \langle w \cdot w \rangle \\ &subject\ to && y_i(\langle w \cdot x_i \rangle + b) \geq 1 \qquad i = 1, ..., l \end{aligned}$$

Here l is the number of training examples, x_i is the input vector, y_i is the desired output. Since directly solving the above optimization problem is difficult, Lagrangian multipliers are used to transform the problem into primal Lagrangian form as below:

$$L(w, b, \alpha) = \frac{1}{2} < w \cdot w > - \sum_{i=1}^{l} \alpha_i [y_i(< w_i \cdot x_i > +b) - 1]$$

where $\alpha_i \geq 0$ are the Lagrange multipliers. The corresponding dual form can be found by differentiating with w and b, and substituting the values obtained for w and b in the primal Lagrangian form:

$$\begin{aligned} \frac{\partial L(w, \alpha, b)}{\partial w} &= w - \sum_{i=1}^{l} y_i \alpha_i x_i && = 0, \\ \frac{\partial L(w, \alpha, b)}{\partial b} &= \sum_{i=1}^{l} y_i \alpha_i && = 0 \end{aligned}$$

The dual form is:

$$L(w, b, \alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j < x_i \cdot x_j >$$

This shows that a hypothesis can be described as a linear combination of of the training points. Another Important point one can observe from the above dual form is, we are computing the dot product between the input vectors, which can be replaced by a kernel function making the SVM solve non-linearly separable problems as well.

This dual form is solved using Karush-Kuhn-Tucker complementary conditions, which state that the optimal solutions α^*, w^*, b^* must satisfy

$$\begin{aligned} \alpha_i y_i(< w_i \cdot x_i > +b) - 1 = 0, && i = 1, ..., l \\ (-1) y_i(< w_i \cdot x_i > +b) - 1 \leq 0, && i = 1, ...l, \\ \alpha_i \geq 0, && i = 1, ...l. \end{aligned}$$

This implies that only those training points which are lie on the optimal hyperplane can only have α_i values positive and all other training points have an α_i value of zero. Training points whose α_i values are greater than zero are called *support vectors*.

However, real world problems are not always linearly separable. Hard margin SVMs fail when training examples are not linearly separable. Errors in training data lead to training examples being mapped to the other side of the hyperplane. Vapnik developed Soft-Margin SVMs to deal with this problem. We shall take up soft-margin SVMs in the next sub-section.

3.3.2 Soft-Margin SVM

One of the problems with hard-margin SVM is that it fails to learn if training data is not linearly separable. Soft-margin SVM tries to overcome this problem by specifying an upper bound on training errors.

$$\begin{array}{lll} minimize_{\xi,w,b} & \langle w \cdot w \rangle + C\sum_{i=1}^{l} \xi_i^2 & \\ subject\ to & y_i(< w \cdot x_i > +b) \geq 1 - \xi_i & i = 1, ..., l \\ & \forall i : \xi_i \geq 0 & \end{array}$$

Here the ξ_i are called slack variables. If training examples lie on the wrong side of the hyperplane, the corresponding ξ_i is greater than 1. Therefore $sum_{i=1}^{n}\xi_i$ is an upper bound on the number of training errors. The factor C is a parameter that allows trade-off between training error and model complexity. A small value of C will increase the number of training errors, whereas a large value of C will make the behaviour closer to hard-margin SVM. The above optimization problems can also be solved as explained in previous section.

3.3.3 Non-Linear SVMs

Many problems in real world inherently have a non-linear structure and linear classifiers are inappropriate. A remarkable property of SVMs is that they can be easily transformed into non-linear learners[35]. In principle, the approach used is as follows. The attribute vectors x_i are mapped onto a high dimensional feature space X using a non-linear mapping $\Phi(x_i)$. The SVM then learns the maximum margin linear classification rule in the feature space X, which may be non linear when projected onto the original input space. In general we compute the dot product in the feature space i.e $\Phi(\overrightarrow{x}_1) \cdot \Phi(\overrightarrow{x}_2)$ rather than finding a mapping like $x \rightarrow \Phi(x)$ which is inefficient to compute. Dot products in feature space can be easily computed using kernel functions $k(x_1, x_2)$, provided the function $k(x_1, x_2)$ satisfies the conditions of Mercer's theorem[35]:

$$\Phi(\overrightarrow{x_1}) \cdot \Phi(\overrightarrow{x_2}) = k(\overrightarrow{x_1}, \overrightarrow{x_2})$$

Depending on the choice of the kernel function, SVMs can learn polynomial classifiers, Radial Basis Function (RBF) classifiers and many others.

$$K_{poly}(\overrightarrow{x_1}, \overrightarrow{x_2}) = (\overrightarrow{x_1} \cdot \overrightarrow{x_2} + 1)^d$$
$$K_{rbf}(\overrightarrow{x_1}, \overrightarrow{x_2}) = exp(-\gamma(\overrightarrow{x_1} - \overrightarrow{x_2})^2)$$

Even though general kernels like polynomial and RBF exist, we have to carefully design a suitable kernel for a given application, using our expertise in the domain. One of the biggest limitation of the support vector approach lies in the choice of the kernel. Another disadvantage with kernel methods is that results are not easily interpretable.

4 Experiments and Results

Test Collection: Empirical evaluations are carried out on the DoE-CIIL corpora of modern Indian languages. We have conducted our experiments on 10 major Indian languages including Assamese, Bengali, Gujarati, Hindi, Kannada, Oriya, Punjabi, Malayalam, Tamil and Telugu. These corpora contain between 250 and 1250 documents. All the corpora are ISCII encoded [36]. The documents are classified into 6 major categories: A-Aesthetics, S-Social Sciences, N-Natural Sciences T-Translated Material, O-Official and Media Language, C-Commerce.

Table 1: Distribution of Documents across Categories in the DoE-CIIL Corpora

Language	No of	Category-wise Breakup					
	Docs	A	S	N	T	O	C
Assamese	1109	387	161	98	39	418	6
Bengali	1268	371	239	133	30	484	11
Gujarati	262	111	60	43	9	3	36
Hindi	1232	449	294	96	44	300	49
Kannada	483	133	214	79	8	5	42
Malayalam	585	122	209	165	26	32	31
Oriya	1139	360	95	136	23	517	8
Punjabi	895	390	166	17	61	261	0
Tamil	747	225	301	140	18	57	6
Telugu	748	281	247	144	30	30	16

It may be observed that the distribution of documents in various categories is not uniform - Commerce for example, includes very few documents. It is also important to note that the category labels given in the corpus are not entirely based on the subject or topic of the documents. *Translated Material* and *Official and Media Language* are not really subject-wise categorizations. For example, we find *literature* as a sub-category under *Aesthetics* as well as *Translated Material*. *Aesthetics* includes *text books* as a sub-category and text books on various topics are included. The document headers that give the major and sub-categories also had several inconsistencies such as spelling variations, format errors, case variations and even completely missing headers. Data preparation was thus a time consuming exercise. The Gujarati corpus was in PC-ISCII and had to be converted to standard 8-bit ISCII.

Initially we develop soft-margin linear SVMs and evaluate the performance. We then compute Mutual Information (MI) between features and categories and reduce the dimensionality of the feature space substantially by including only the more promising features. All further explorations are done on the reduced feature set. We see that SVMs out-perform naive Bayes and kNN methods. We also see that non-linear SVMs using Polynomial and Radial Basis Function kernels do not perform any better than the simpler linear soft-margin SVMs. We have used the SVM^{light} tool developed by T.Joachims. Naive Bayes and kNN algorithms have been implemented by us here using Perl under Linux.

We have used one-vs.-all SVM for multi-classification. SVMs are binary classifiers. One SVM is therefore built for each category. A given test document is tested by each of the SVMs and any or all of them may classify the document positively for the respective categories. A document is taken as having been classified if one and only one SVM classifies it positively and all the others negatively. Performance is reported in terms of Precision and Recall as also using the combined F measure:

$$P = \frac{No.\quad of\quad Documents\quad correctly\quad classified}{No.\quad of\quad documents\quad classified}$$
$$R = \frac{No.\quad of\quad Documents\quad correctly\quad classified}{No.\quad of\quad documents}$$
$$F = \frac{2PR}{P+R}$$

3-fold cross-validation (CV) has been carried out in all cases. (Since the number of documents available in each category is limited, it would not be practicable to perform more fine grained cross-validation such as 10-fold cross-validation.) We give the results of our soft-margin linear SVMs in the Table 2.

Table 2: Performance of Soft-Margin Linear SVM for DoE-CIIL corpora

Language	# Features	P%	R%	F%
Assamese	185,341	85.04	63.33	72.60
Bengali	182,464	87.44	64.08	73.96
Gujarati	105,785	86.70	53.29	66.01
Hindi	124,930	84.73	55.66	67.19
Kannada	344,296	87.40	60.40	71.44
Malayalam	535,960	88.60	54.75	67.68
Oriya	152,432	92.09	78.87	84.97
Punjabi	105,502	82.13	51.48	63.29
Tamil	449,337	83.20	60.63	70.15
Telugu	623,266	90.01	60.53	72.39

Taking surface words as features in Indian languages leads to a very high dimensional feature space. Not all features may be significant for classification. We have used

MI for feature selection. MI measures how much a given word is relevant to a given category. For each category we have ranked words on the computed MI measure. We have selected top ranked words using a threshold. Optimal thresholds are found for each languages by extensive experimentation. We give in Table 3 the optimum thresholds obtained, the reduced number of features for each language and performance after dimensionality reduction.

Table 3: Comparison of results after Dimensionality Reduction with MI for DoE-CIIL corpora

Language	No. of Features	F%	Threshold	Reduced # Features	P%	R%	New F%
Assamese	185,341	72.60	0.00001	41,343	83.13	67.23	76.28
Bengali	182,464	73.96	0.00001	33,603	87.11	74.67	80.41
Gujarati	105,875	66.01	0.00005	53,689	88.50	70.53	78.50
Hindi	124,930	67.19	0.000008	37.213	82.76	74.89	78.63
Kannada	344,296	71.44	0.00005	47,519	83.02	74.66	79.62
Malayalam	535,960	67.68	0.0012	18,203	88.91	72.05	79.06
Oriya	152,432	84.97	0.000005	32,194	91.14	83.47	87.14
Punjabi	105,502	63.29	0.00001	29,318	82.80	71.14	76.53
Tamil	449,337	70.15	0.00003	37,476	80.08	74.04	76.97
Telugu	623,266	72.39	0.00001	55,638	86.33	75.19	80.38

It may be observed that the performance has not decreased, rather it has improved substantially, despite heavy reductions in the feature dimensionality. This is due to elimination of many terms which actually act as noise rather than as discriminating features. Manual observation of features corroborates with this intuition. With reduced feature space, computational complexity is also naturally reduced.

Although all SVMs can in principle classify a document positively, we have found in our experiments here that at most two SVMs have classified a document positively. This happens about 20% of the cases on the average and in every one of these cases, the correct class is invariably included.

Table 4: Confusion Matrix for all languages of Doe-CIIL corpora

Category	A	S	N	T	O	C
A	1989	111	25	4	87	0
S	147	1281	41	0	47	3
N	27	62	756	0	17	0
T	88	32	6	16	5	0
O	136	26	11	6	1499	0
C	0	41	12	0	1	75

It can be seen from the confusion matrix (Table4) that certain categories such as *Translated Material* show maximum confusion. The categories given in the DoE-CIIL corpora are not entirely subject or topic based. *Aesthetics* is main category as also a sub-category both under *Translated Material*. *Administration* is a sub-category of both *Social Science* and *Translated Material*. If we were to look from the point of actual subject-wise categories, the classification given by our system will be much better than the performance measures indicated in the tables here.

We have also experimented with two other well known techniques used for text categorization namely naive Bayes and kNN. A range of k values (3,7,15,30) have been tried. We have used the cosine similarity measure. In the case of naive Bayes classifier, we have not used any thresholding for incorporating a reject option. Thus Precision, Recall and F-Measure will all be same. In the case of kNN also, simply the majority class is assigned and so the Precision, Recall and F-Measure will all be the same. We compare the best results obtained with the soft-margin linear SVMs in Table 5.

Table 5: Comparison of NB, kNN and SVM Classifiers for DoE-CIIL corpora

Language	NB F%	kNN F%	SVM F%
Assamese	58.52	64.20	76.28
Bengali	69.72	69.53	80.41
Gujarati	71.40	77.27	78.50
Hindi	68.90	66.31	78.63
Kannada	69.73	75.25	79.62
Malayalam	70.30	76.41	79.06
Oriya	74.47	77.90	87.14
Punjabi	60.02	70.10	76.53
Tamil	62.95	73.89	76.97
Telugu	72.90	75.60	80.38

We see that soft-margin SVMs out-perform other techniques.

We have also experimented with kernel based SVMs. We have explored Polynomial and Radial Basis Function kernels. The results below show that non-linear SVMs do not give any significant improvements over the simpler linear soft-margin SVMs.

4.1 Experiments on a Different Corpus

In order to check for corpus effects, we have experimented with a different corpus as well [5]. The second corpus we have used is the LERC-UoH Telugu corpus, a nearly 40 Million word text corpus developed by us at University of Hyderabad [3]. In our experiments here, we use only a part of this corpus containing the iinaaDu newspaper articles. The corpus was developed by downloading the articles from iinaaDu newspaper between July 2003 and March 2004 and converting the font-encoded pages into ISCII standard using tools developed by us. The corpus includes more than 9500 articles totaling to about 26 Million words. Of this, 794 documents in 4 major categories (P-Politics, S-Sports, B-Business, and C-Cinema) have been used in the current set of experiments. The distribution of the documents across these four categories for each language is tabulated below.

Table 6: Comparison of RBF and Polynomial kernels with linear SVM for DoE-CIIL corpora

Language	SVM with RBF			SVM with Polynomial			Linear SVM
	g=0.6	g = 0.8	g=1	d=1	d=2	d=3	F%
Assamese	61.52	54.9	52.05	75.28	74.10	73.27	76.28
Bengali	77.98	76.22	74.92	79.48	78.22	78.21	80.41
Gujarati	70.91	71.38	71.38	74.91	70.30	71.85	78.50
Hindi	68.39	65.83	63.17	78.63	78.69	78.69	78.63
Kannada	71.39	67.67	63.15	78.60	75.12	70.87	79.62
Malayalam	75.55	72.96	71.84	75.75	72.72	70.28	79.06
Oriya	86.22	86.05	86.01	87.14	86.84	85.79	87.14
Punjabi	75.92	75.92	75.82	76.53	76.28	76.11	76.53
Tamil	76.56	76.33	75.81	74.94	72.95	72.10	76.97
Telugu	76.22	74.69	74.90	79.01	75.23	73.72	80.38

Table 7: Distribution of Documents across Categories in the LERC-UoH Telugu News Paper Corpus

Total	Category-wise Breakup			
	Politics	Sports	Business	Cinema
794	307	205	189	93

We have carried out experiments using soft margin SVM with 3-fold cross validation, applied MI for feature reduction, found optimal feature set, and then compared the performance of soft-margin SVM with naive Bayes and kNN on the reduced set of features. See results in Table 8.

Table 8: Comparison of performances of naive Bayes, kNN and soft-margin SVM after dimensionality reduction on LERC-UoH Telugu newspaper Corpus

Language	# Features	SVM F%	T	Reduced No. of Features	SVM F%	NB F%	kNN F%
Telugu	171,996	95.05	0.00001	18,930	96.39	91.70	93.47

Again we see that simple linear soft-margin SVMs are the best.

5 Conclusions

This paper is about Text Categorization in Indian languages. After a clear formulation of the Text Categorization task in its various dimensions, we have briefly described the complexities involved in Indian languages. We have included the results of our experiments on 10 major Indian languages using purely corpus bases statistical techniques. No linguistic expertise in the languages concerned is necessary. We have developed soft-margin SVMs and we demonstrate that they out-perform other techniques including kNN, naive Bayes and even kernel based non-linear SVMs. We see that the results are similar when applied to a different corpus with a different set of categories as well. We have also shown that Mutual Information is an effective language independent dimensionality reduction technique. Even after substantial reductions in the dimensionality of the feature space, classification performance is not reduced but rather increased, thanks to the reduction in noise terms. Results obtained are good despite the fact that Indian languages are morphologically very complex and no morphological analyzer or stemmer has been used. Further improvements in performance may be possible as and when adequate lexical resources and morphological analyzers etc. are fully developed for Indian languages. We plan to work with more fine grained categories and also look at hierarchical classification in the near future.

References

[1] Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by latent semantic analysis. Journal of the American Society of Information Science **41**(6) (1990) 391–407

[2] Murthy", K.N.: "Natural Language Processiong - an Information Access Perspective". "Ess Ess Publications" (2006)

[3] Kumar, G.B., Murthy, K.N., Chaudhuri, B.: Statistical analysis of Telugu text corpora. IJDL, Vol 36, No 2 (June 2007)

[4] Murthy, K.N.: Universal Clause Structure Grammar. PhD thesis, Department of CIS, University of Hyderabad (1996)

[5] Murthy, K.N.: Automatic text categorization. In: International Symposium on Linguistics, Qualification and Computation, Hyderabad, INDIA (2005) 9–11

[6] Negi, A., Murthy, K.N., Bhagvati, C.: Foundational issues of document engineering in Indian scripts and a case study in telugu. Vivek **16**(2) (2006) 2–7

[7] N Fuhr et al: Air/x—a rule-based multistage indexing system for large subject fields. In: Proceedings of RIAO 91. (1991) 606–623

[8] Yang, Y., Chute, C.G.: An example-based mapping method for text categorization and retrieval. In: ACM Transaction on Information Systems:. (1994) 253–277

[9] Masland, B., Linoff, G., Waltz, D.: Classifying news stories using memory based reasoning. In: Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, US (1994) 81–93

[10] Yang, Y.: Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In: Proceedings of ACM SIGIR. (1994) 13–22

[11] Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In Fisher, D.H., ed.: Proceedings of ICML-97, 14th International Conference on Machine Learning, Nashville, US, Morgan Kaufmann Publishers, San Francisco, US (1997) 412–420

[12] Yang, Y.: An evaluation of statistical approaches to text categorization. In: Journal of Information Retrieval. (1999) 69–90

[13] W Lam and C Y Ho: Using a generalized instance set for automatic text categorization. In: Proc. of the 21th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98). (1998) 81–89

[14] Tzeras, K., Hartmann, S.: Automatic indexing based on Bayesian inference networks. In Korfhage, R., Rasmussen, E., Willett, P., eds.: Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval, Pittsburgh, US, ACM Press, New York, US (1993) 22–34

[15] Lewis, D.D., Ringuette, M.: A comparison of two learning algorithms for text categorization. In: Proceedings of SIGIR 92. (1992) 59–65

[16] Moulinier, I.: Is learning bias an issue on the text categorization problem? Technical report, Universite Paris VI (1997)

[17] Koller, D., Sahami, M.: Hierarchically classifying documents with very few words. In: 14th Int. Conf. on Machine Learning ICML'97. (1997) 170–178

[18] Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In Nédellec, C., Rouveirol, C., eds.: Proceedings of ECML-98, 10th European Conference on Machine Learning, Chemnitz, DE, Springer Verlag, Heidelberg, DE (1998) 137–142

[19] McCallum, A., Nigam, K.: A comparison of event models for naive Bayes text classification. In: AAAI-98 Workshop on Learning for Text Categorization. (1998)

[20] Baker, L.D., Mccallum, A.K.: Distributional clustering of words for text categorization. In: Proceedings of the 21th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98). (1998) 96–103

[21] Apte, C., Damerau, F., , Weiss, S.: Text mining with decision rules and decision trees. In: Proc. of Conf. on Automated Learning and Discovery. (1998)

[22] Apte, C., Damerau, F., Weiss, S.M.: Towards language independent automated learning of text categorization models. In: Proceedings of 17th Annual ACM/SIGIR conference. (1994)

[23] Cohen, W.W.: Text categorization and relational learning. In Prieditis, A., Russell, S.J., eds.: Proceedings of ICML-95, 12th Int. Conf. on Machine Learning, Morgan Kaufmann Publishers, San Francisco, US (1995) 124–132

[24] Cohen, W.W., Singer, Y.: Context-sensitive learning methods for text categorization. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. (1996) 307–315

[25] Moulinier, I., Raskinis, G., , Ganascia., J.G.: Text categorization: a symbolic approach. In: Proceedings of SDAIR-96, Annual Symposium on Document Analysis and Information Retrieval. (1996)

[26] Wiener, E., Pedersen, J.O., Weigend., A.S.: A neural network approach to topic spotting. In: Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval. (1995) 317–332

[27] Ng, T.H., Goh, W.B., Low, K.L.: Feature selection, perceptron learning, and a usability case study for text categorization. In Belkin, N.J., Narasimhalu, A.D., Willett, P., eds.: Proceedings of SIGIR-97, 20th ACM International Conference on Research and Development in Information Retrieval, Philadelphia, US, ACM Press, New York, US (1997) 67–73

[28] Lewis, D.D., Schapire, R.E., Callan, J.P., Papka, R.: Training algorithms for linear text classifiers. In Frei, H.P., Harman, D., Schäuble, P., Wilkinson, R., eds.: Proceedings of SIGIR-96, 19th ACM Int. Conf. on Research and Development in Information Retrieval, Zürich, CH, ACM Press, New York, US (1996) 298–306

[29] Joachims, T.: Learning To Classifying Text Using Support Vector Machines. Kluwer Academic Publishers (2001)

[30] Yang, Y., Liu, X.: A re-examination of text categorization methods. In Hearst, M.A., Gey, F., Tong, R., eds.: Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval, Berkeley, US, ACM Press, New York, US (1999) 42–49

[31] Dasarathy, B.V.: Nearest neighbour (nn) norms: Nn pattern classification technique. IEEE Computer Society Press (1991)

[32] Iwayama, M., Tokunaga, T.: Cluster-based text categorization: a comparison of category search strategies. In Fox, E.A., Ingwersen, P., Fidel, R., eds.: Proceedings of SIGIR-95, 18th ACM Int. Conf. on Research and Development in Information Retrieval, Seattle, US, ACM Press, New York, US (1995) 273–281

[33] Vapnik, V.N.: Statistical Learning Theory. Wiley-Interscience Publication (1998)

[34] Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press (2000)

[35] Cristianini, N., Shawe-Taylor, J.: Kernel Methods for Pattern Analysis. Cambridge University Press (2002)

[36] Beareau of Indian Standards: ”indian script code for information interchange - ISCII”. In: ”IS13194: 1991”, New delhi, India (1991)