Discriminative Models for Information Retrieval

Ramesh Nallapati
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
nmramesh@cs.umass.edu

ABSTRACT

Discriminative models have been preferred over generative models in many machine learning problems in the recent past owing to some of their attractive theoretical properties. In this paper, we explore the applicability of discriminative classifiers for IR. We have compared the performance of two popular discriminative models, namely the maximum entropy model and support vector machines with that of language modeling, the state-of-the-art generative model for IR. Our experiments on ad-hoc retrieval indicate that although maximum entropy is significantly worse than language models, support vector machines are on par with language models. We argue that the main reason to prefer SVMs over language models is their ability to learn arbitrary features automatically as demonstrated by our experiments on the home-page finding task of TREC-10.

Categories and Subject Descriptors

H.3.3 **[Information Search and Retrieval]**: Retrieval models

General Terms

Algorithms, Theory, Experimentation

Key words

Pattern classification, machine learning, discriminative models, maximum entropy, support vector machines.

1. INTRODUCTION

Traditionally, information retrieval models have viewed the problem of retrieval as one of measuring the similarity between the documents and queries. For example, the vector space model [28] considers documents and queries as vectors in term-space and measures the similarity of the document to the query by the cosine of the angle between the two vectors. However, one of the shortcomings of the vector-space model is that term-weights are empirically tuned and the model provides no theoretical basis for computing optimum weights.

SIGIR'04, July 25–29, 2004, Sheffield, South Yorkshire, UK.

One of the first theoretically motivated IR models is the binary independence retrieval (BIR) model introduced by Robertson and Sparck Jones [25] in 1976. To the best of our knowledge, this is the first model that viewed IR as a classification problem. They consider retrieval as essentially a process of classifying the entire collection of documents into two classes: relevant and non-relevant. However, instead of doing a hard classification, they estimate the probability of relevance and non-relevance with respect to the query and rank the retrieved documents by their log-likelihood ratio of relevance. Although this was a promising framework, the model did not perform well because of problems in estimation of probabilities.

Ignoring the details of estimation, we subscribe to Robertson's view of IR as a problem of binary classification of relevance. We believe this view has certain inherent advantages. Firstly, this framework mirrors the real-life IR process very accurately: a user is primarily concerned about how relevant a given document is to his(her) information need and the model aims at quantifying exactly that. Secondly, casting IR as a binary classification process allows us to leverage many sophisticated techniques developed in the machine learning domain.

In the recent past, a class of techniques called discriminative models have enjoyed good empirical success in many applications of machine learning. In this work, we explore the applicability of discriminative classifiers to IR, regarding the problem as one of binary classification.

The rest of the paper is organized as follows. In section 2 we discuss the modeling differences between generative and discriminative models and then show that the existing probabilistic models for IR are generative models. In section 3 we argue the case for discriminative models for IR. We present an overview of maximum entropy models and support vector machines, two discriminative models that we used in the current work and other modeling issues in sections 4 and 5. Some of the experiments conducted and results obtained are presented in section 6. We then compare our work with other related work in section 7. Section 8 concludes this paper with a few remarks on future directions.

2. DISCRIMINATIVE AND GENERATIVE CLASSIFIERS

Pattern classification is an important problem in machine learning and can be defined as the problem of classifying an example based on its vector of features $\mathbf{x}$ into its class C. Classification models typically map an example to its class through a discriminant function that can be a posterior probability given by $P(C|\mathbf{x})$ or simply a confidence score $g(C|\mathbf{x})$. In the sub-field of supervised learning, a classifier learns its discriminant function from a set of labeled training examples. There are several existing pattern classi-

fication techniques in the literature and they can be broadly classified into two major categories: generative models and discriminative models. Algorithms that model the posterior $P(C|\mathbf{x})$ directly or learn a direct map from inputs $\mathbf{x}$ to the class labels C given by $g(C|\mathbf{x})$ are called discriminative models. Generative models, on the other hand, model the class-conditional probability $P(\mathbf{x}|C)$ and the prior probability $P(C)$ and estimate the posterior through the application of Bayes' rule as shown below.

$$P(C|\mathbf{x}) \propto P(\mathbf{x}|C)P(C) \tag{1}$$

In the field of IR, the most famous formal models are the BIR model, the Two-Poisson model and the current state-of-the-art language models. We will now examine these models in more detail and see if we can label these models into one of these broad categories.

2.1 Probabilistic IR models as Classifiers

2.1.1 Binary Independence Retrieval (BIR) model

As discussed in section 1, the BIR model views IR as a problem of binary classification. The relevant class is represented by R and the non-relevant class by $\bar{R}$. They define the feature vector $\mathbf{D}$ of an example D (in this case, a document) as $\mathbf{D} = (x_1, x_2, \cdots, x_n)$ where n is the number of terms in the vocabulary and x_i is a binary random variable indicating the presence or absence of the i^{th} index term. Ranking is done by the log-likelihood ratio of relevance which in turn is evaluated using Bayes' rule as shown below:

$$\log(\frac{P(R|\mathbf{D})}{P(\bar{R}|\mathbf{D})}) = \log(\frac{P(\mathbf{D}|R)}{P(\mathbf{D}|\bar{R})}\frac{P(R)}{P(\bar{R})}) \tag{2}$$

$$= \log(\prod_{i:x_i=1}\frac{P(x_i=1|R)}{P(x_i=1|\bar{R})}\prod_{i:x_i=0}\frac{P(x_i=0|R)}{P(x_i=0|\bar{R})}\frac{P(R)}{P(\bar{R})}) \tag{3}$$

where equation 3 follows from the assumption of conditional independence.

From equation 2, it is easy to infer that the BIR model belongs to the class of generative models. Despite a promising framework, the model has not met with good empirical success owing to the difficulty in estimating the class conditional $P(x_i = 1|R)$. Since we do not know the set R in the beginning, they assume uniform probability distribution over the entire vocabulary and update the probabilities as relevant documents are provided by the user. Thus the model depends largely on user-relevance-feedback to compute accurate ranking of relevant documents.

2.1.2 Two-Poisson model

This model follows exactly the same framework as that of the BIR model, but instead of a multiple Bernoulli distribution, they use a mixture of two Poisson distributions to model the class conditionals $P(\mathbf{D}|R)$ and $P(\mathbf{D}|\bar{R})$. One of the Poissons called the elite class E, is expected to generate content bearing words while the non-elite class $\bar{E}$ generates non-informative words in the document [9] as shown below.

$$\begin{aligned} P(D|R) &= \prod_{i=1}^{n}(P(c(w_i)=k|E)P(E|R) + \\ &\quad P(c(w_i)=k|\bar{E})P(\bar{E}|R)) \qquad (4) \\ &= \prod_{i=1}^{n}(p\frac{\exp(-l)l^k}{k!} + (1-p)\frac{\exp(-m)m^k}{k!}) \qquad (5) \end{aligned}$$

where l and m are the parameters of the two Poissons and $p = P(E|R)$ is the mixture weight. The assumption here is that given the eliteness of a term, its count k in the document is conditionally independent of the document's relevance. They estimate $P(D|\bar{R})$ on similar lines and finally use equation 2 to compute the score. Clearly, the Two-Poisson also is a generative model. Similar to the BIR model, it also needs relevance feedback for accurate parameter estimation [27].

2.1.3 Language models

More recently, in 1998, Ponte and Croft [23] proposed the language modeling approach to information retrieval. In this approach, each document is assigned a language model which is a probability distribution over terms. Accordingly, the ranking of a document is given by the probability of generation of the query from document's language model as shown below.

$$P(Q|D) = P(q_1, q_2, .., q_n|D) = \prod_{i=1}^{n} P(q_i|D) \tag{6}$$

where the last term in the above equation is obtained from the assumption of conditional independence of terms given the document's language model. Since the document is typically larger than a query, estimating a document's model is an easier task than estimating a model of relevant documents from a query. Thus, through a clever modeling variation, language models have managed to circumvent the problem of estimating the model of relevant documents that the BIR model and Two-Poisson suffer from.

Language models have been quite successful in several IR tasks and their performance has been shown to be on par with the vector space model. Since language models also offer a formal framework for IR, a lot of interest has been generated in the last few years on applying language models and its extensions to several IR related tasks.

The classification of language models into one of discriminative or generative classifiers does not seem immediately obvious. In fact, language models appear to have abandoned the notion that IR is a binary classification problem. In the original version as presented in [23], there is no reference to the class variable R that denotes relevance or non-relevance. However, if we imagine each document's model D as a unique class and the task of IR to be that of classifying a query into its best class as given by the posterior $P(D|Q)$, then Bayes' rule tells us that this can be estimated as shown below.

$$P(D|Q) \propto P(Q|D)P(D) \tag{7}$$

If we assume a uniform prior $P(D)$ over all documents, then the posterior depends entirely on the conditional $P(Q|D)$ which is the scoring function used by language models. Hence in this view, language models can be considered generative classifiers in a multi-class classification sense. This view is somewhat peculiar in the sense that language models essentially rank the classes (documents) for each instance (query).

3. THE CASE FOR DISCRIMINATIVE MODELS FOR IR

Although generative models such as naïve Bayes' and hidden Markov models have been traditionally popular in the pattern classification community, discriminative models (eg. Maximum Entropy, SVMs, conditional random fields) have been preferred over generative models in the recent past in many domains. One of the reasons for using discriminative models rather than generative models, as articulated by Vapnik [30], is that "one should solve the (classification) problem directly and never solve a more general problem (class-conditional) as an intermediate step". There has

also been some empirical evidence that discriminative models tend to have a lower asymptotic error as the training set size is increased [19].

Apart from theoretical considerations, we believe there are reasons specific to the domain of IR that make discriminative models suitable to the task. The remaining part of the sections explains a few of them.

3.1 Modeling assumptions

Like most generative models, language models for IR make certain modeling assumptions that are not strictly valid. For example, they assume that terms are conditionally independent given the document's model. From our own understanding of natural language, we know that the assumption of term independence is a matter of mathematical convenience rather than a reality. Similarly, language models assume documents obey a multinomial distribution of terms. It has been shown in [29] that multinomial distribution fails to model the burstiness of terms, i.e., for a given number of occurrences of a term, the probability of multiple occurrences is much higher than what a multinomial model would expect. Discriminative models on the other hand, typically make very few assumptions and in a sense, let the data speak for itself.

3.2 Expressiveness

Many search engines provide advanced search options to take into account situations of information need where relevance is determined by other types of query-based features in addition to term frequencies. For example, one might want to assign high importance to proximity of query terms. For some queries, ordering of the query terms may be critical as in noun-phrase queries and for others, exact matching is required. In some other cases, we want to return documents that contain some query terms but do not contain others. Language models are not expressive enough to incorporate such features into the model. Attempts have been made to incorporate proximity based features in language models [18], but doing so requires many modeling assumptions and is often unwieldy. Discriminative models on the other hand are not plagued by such limitations and one can include all such features effortlessly into a single model.

3.3 Learning arbitrary features

A related desirable feature in a modern IR model is its ability to learn automatically a host of arbitrary features that characterize present-day collections. For example, in some domains like the web or scientific literature, there are different representations of the document such as document content, anchor-text, title and abstract. There may also be other query-independent features that influence relevance such as popularity of the document. Moreover, we know that relevance is a subjective matter that depends to a large extent on the user's personal preferences and background. A next-generation IR model should be able to learn such features automatically from labeled examples.

Language models do permit combining different document representations using mixture models [21] whose weights can be learnt automatically using the EM algorithm. Some work has also been done to include query-independent features into the prior probability (see equation 7) of the document [14]. However, the weights of these features are determined by empirical means.

In view of the many query dependent and query-independent document features and user-preferences that influence relevance, we believe that a discriminative model that learns all the features automatically in a unified manner, making the least number of assumptions, is best suited for the generalized IR problem.

3.4 Notion of relevance

In language modeling, there is no explicit notion of relevance, as we have seen earlier in section *2.1.3*. There has been considerable controversy on the missing relevance variable in language models (see [26] for an interesting discussion). In response to this discussion, Lafferty and Zhai [15] argued that relevance is an implicit variable in language modeling and what we are actually estimating is $P(Q|D,R)$.

We believe that Robertson's view of IR as a binary classification problem of relevance is more realistic than the implicit notion of relevance as it exists in language models. As we have discussed in our opening remarks in section 1, explicit modeling of relevance helps quantify the extent to which a user's information need is satisfied.

4. MODELS USED IN CURRENT WORK

In this work, we consider two popular discriminative models, the maximum entropy approach [1] and support vector machines [2] for the IR-problem. Both techniques have been successfully applied in several natural language processing tasks such as tagging [24] and text-classification [20, 12]. We summarize the two methods very briefly below.

4.1 Maximum entropy (ME) model

Intuitively, the principle of maximum entropy is simple: model all that is known and assume nothing about that which is unknown. In other words, given a collection of facts, choose a model consistent with all the facts, but otherwise as uniform as possible [1]. Accordingly, the model seeks to maximize the entropy of the posterior conditional distribution $P(R|D)$ subject to the constraint that the expected values of certain feature functions as predicted by the model should comply with their corresponding empirical frequencies observed in a training set.

The parametric form of the maximum entropy probability function can be expressed as follows:

$$P(R|D,Q) = \frac{1}{Z(Q,D)} \exp(\sum_{i=1}^{n} \lambda_{i,R} f_i(D,Q)) \qquad (8)$$

Note that we used the notation $P(R|D,Q)$ instead of $P(R|D)$ to make it explicit that the relevance of the document is measured with respect to the specific query. Here, $Z(Q,D)$ is a normalizing constant, $f_i(D,Q)$ are the feature functions of the document with weights $\lambda_{i,R}$ and n is the number of features. The feature weights are learned from training data using a fast gradient descent algorithm [17].

As in Robertson's BIR model, we use the log-likelihood ratio as the scoring function for our ranking as shown below.

$$\log \frac{P(R|D,Q)}{P(\bar{R}|D,Q)} = \sum_{i=1}^{n} (\lambda_{i,R} - \lambda_{i,\bar{R}}) f_i(D,Q) \qquad (9)$$

4.2 Support Vector Machines (SVM)

The discriminant function in support vector machines is given by the hyper-plane that separates the two classes of training examples with the largest margin [2]. It is expected that larger the margin, better is the generalization of the classifier. The hyper-plane is in a higher dimensional space called kernel space and is mapped from the feature space. The mapping is done through kernel functions that allow us to operate in the input feature-space while providing us the ability to compute inner products in the kernel space. The key idea in mapping to a higher space is that, in a sufficiently high dimension, data from two categories can always be separated by

a hyper-plane. Thus if $\mathbf{f}(D, Q)$ is the vector of features, then the discriminant function is given by

$$g(R|D, Q) = \mathbf{w} \bullet \phi(\mathbf{f}(D, Q)) + b \quad (10)$$

where $\mathbf{w}$ is the weight vector in kernel space that is learnt by the SVM from the training examples, $\bullet$ denotes inner product, b is a constant and ϕ is the mapping from input space to kernel space. The equation $g(R|D, Q) = 0$ represents the equation of the hyper-plane in the kernel space. The value of the discriminant function $g(R|D, Q)$ for an arbitrary document D and a query Q is proportional to the perpendicular distance of the document's augmented feature vector $\phi(\mathbf{f}(D, Q))$ from the separating hyper-plane in the kernel space. The SVM is trained such that $g(R|D, Q) \geq 1$ for positive(relevant) examples and $g(R|D, Q) \leq -1$ for negative (non-relevant) examples as long as the data is separable. Thus, higher the value of the discriminant function, more is our confidence that the document is relevant. Note that the values returned by SVMs are not probabilities but still represent and quantify the degree of relevance.

Note that both discriminative models, while retaining the basic framework of the BIR model, avoid estimating the class-conditional (that landed the BIR model in estimation difficulties) and instead directly compute the posterior $P(R|Q, D)$ or the mapping function $g(R|D, Q)$ from the feature vectors of query-document pairs.

5. OTHER MODELING ISSUES

There are a few important modeling issues that are specific to discriminative models and the IR task. Before we proceed to reporting our experiments, we present those issues in detail here.

5.1 Out of Vocabulary words (OOV) problem

Although in this work we view IR as a problem of binary classification, it still remains distinct from other classification tasks such as text classification. In text-classification for example, the features are words and feature values are typically their frequencies. Based on a set of training examples, the classifier learns the words that are most discriminative in determining the class and weights them accordingly. The classifier then assigns labels to test examples based on their word-based features. If a test example contains out-of-vocabulary (OOV) words, words that are unseen in the training set, the classifier typically ignores them. OOV words may not be critical in text classification, but would prove disastrous in an IR classifier based on word-features: test queries are almost always guaranteed to contain words that are not seen in the training queries. Hence, we define our features not based on words themselves, but on query-based statistics of documents such as the total frequency of occurrences of all query terms in the document or the sum-total of the idf-values of the query terms that occur in the document. Note that this does not compromise on the expressiveness of the discriminative models in any way. The actual feature functions depend on the task at hand as we will in see in the following sections.

5.2 Unbalanced data

The IR problem is characterized by *unbalanced* training sets in which one of the classes (non-relevant) is represented by a large portion of all the examples, while the other (relevant) class has only a small percent of the examples. When dealing with unbalanced class distributions, discriminative algorithms such as SVM that maximize classification accuracy result in trivial classifiers that completely ignore the minority class. Some of the typical methods of dealing with this problem include oversampling the minority class by repeating minority examples, under-sampling the majority class or adjusting misclassification costs. Oversampling minority class or adjusting misclassification costs involves dealing with all the existing majority class (non-relevant) examples and may result in considerable computational costs during training in tasks such as IR that have an overwhelmingly large number of majority class examples. Hence we chose the method of under-sampling the majority class, in which we sample examples from majority class such that the number of training examples in both classes are made equal. While there are several recommendations in the literature on how to sample, we chose the method of random sampling as there is some empirical evidence that this method does better than or as well as the other existing ones [32].

6. EXPERIMENTS AND RESULTS

Our experimental work consists of two parts. In the first part, our aim is to compare and contrast the performance of the discriminative models with that of generative models on the core IR task of ad-hoc retrieval. Since language modeling is the best performing generative model till date, we used it as our generative model baseline in our experiments. As our representative discriminative models, we used the maximum entropy model and SVMs.

In the second part, we run the discriminative models on the web-task of home-page-finding where a variety of features influence relevance. This experiment is aimed at demonstrating the power of discriminative models in learning multiple query-dependent and query-independent features automatically.

6.1 Ad-hoc retrieval

In this task, we compare our runs on 4 different TREC collections. The details of the collections used and their statistics are shown in figure 1. All the collections are fairly large, running into hundreds of thousands of documents and on the order of a gigabyte in index size. Our preprocessing steps in creating an index include stemming using the *K-stemmer* and removing stop-words. For each collection, we chose a set of training and testing queries for which relevance judgments are available, as shown in the same figure. We used only title queries since they are the most reflective of real-life queries a user would typically issue to a search engine.

For each collection, we train our models using the training set of queries and the corresponding labeled relevant and non-relevant documents. We run this trained model on the test-queries of all four collections and measure average precision. In all we have 16 train-test combinations corresponding to the 4 collections.

In our language model runs, we use Dirichlet smoothing as it has been shown to be the most effective smoothing on short queries [31]. Training the language model consists of learning the optimal value of the smoothing parameter. We did this empirically by performing an extensive parameter search and setting the value of the Dirichlet parameter to the one that gives the best average precision on the training set of queries. All our language model runs were performed using *Lemur* [34].

Training the discriminative models consists of providing a set of labeled examples with a list of features to the learning algorithm. We used only statistics such as *tf* and *idf* and their combinations as features as shown in figure 2. We believe this provides for a fair comparison of these models with the language modeling baseline which relies on such statistics alone. Note that the aim of the current set of experiments is only to test the robustness of the discriminative models. Their ability to learn additional features automatically is tested in our experiments reported in section 6.2. An IR expert would immediately notice that some of the features were defined so as to capture the standard *tf-idf* metric used in vector

	Disks 1-2	Disk 3	Disks 4-5	WT2G
Num. Docs	741,856	336,310	556,077	247,491
Num. Terms	177,577,452	83,358,487	181,236,183	159,777,896
Num. Unique Terms	612,627	386,033	721,571	1,235,083
Avg. Doc. length	239	247	325	645
Index size	1.51GB	0.71GB	1.53GB	1.38GB
Training queries	101-150	51-100	301-350	401-425
Test queries	151-200	101-150	401-450	426-450

Figure 1: Statistics of various collections used in our ad-hoc retrieval experiments

	Feature		Feature
1	$\sum_{q_i \in Q \cap D} log(c(q_i, D))$	4	$\sum_{q_i \in Q \cap D} (log(\frac{\|C\|}{c(q_i, C)}))$
2	$\sum_{i=1}^{n} log(1 + \frac{c(q_i, D)}{\|D\|})$	5	$\sum_{i=1}^{n} log(1 + \frac{c(q_i, D)}{\|D\|} idf(q_i))$
3	$\sum_{q_i \in Q \cap D} log(idf(q_i))$	6	$\sum_{i=1}^{n} log(1 + \frac{c(q_i, D)}{\|D\|} \frac{\|C\|}{c(q_i, C)})$

Figure 2: Features in the discriminative models: $c(w, D)$ represents the raw count of word w in document D, C represents the collection, n is the number of terms in the query, $|.|$ is the *size-of* function and $idf(.)$ is the inverse document frequency.

space as well as language models. We used the *log* function in the features to dampen the effects of large numbers. Note that features numbered 4 and 6 are obtained by replacing *idf* in features 3 and 5 respectively by the inverse of the collection frequency. Since language models use this frequency in smoothing, we thought it fit to include these features.

We used *svm-light*[16] for our SVM runs and the toolkit of Zhang [33] for our ME runs. Although we do not report here, we have tried other variations of frequency based features but we found that the ones we report give just about the best performance. We also noticed that using document length or any of its normalized variations as extra features actually hurt the performance. Note that language models do not use document length as an explicit feature but use it only to determine the extent of smoothing as in Dirichlet prior[31]. We have also tried various versions of the SVM kernel functions including higher order polynomials and radial basis functions. We found that linear kernels give the best performance on most datasets. Although quadratic kernels do marginally better on some collections, we felt it was not a good bargain compared to the higher computational costs involved in training them. Linear kernels on the other hand are well behaved in their training and converge very rapidly on all collections. Using linear kernels means that SVMs draw the separating hyper-plane in the original feature space. In a sense, it is a disappointing result since the power of SVMs lies in projecting the data to a higher dimensional space. Nevertheless, we believe that the other property of the SVMs, namely maximization of the margin, makes the SVMs retain their attractiveness to IR.

As described in section 5.2, we used random-under-sampling of the non-relevant class to deal with the unbalanced data problem. Since each random sampling of training examples gives rise to a distinct model, we repeated each run of both discriminative models five times using a new random sample each time and computed an average value of the average precision.

Figure 3 shows the comparison of the performances of language model(LM), support vector machines (SVM) and maximum entropy (ME) models for each of the 16 train-test combinations described above. In 50% (8/16) of the runs the performance of SVM is statistically indistinguishable from that of the LM while in 12.5 % (2/16) of the cases, it is statistically better than that of LM (using paired two-tailed T-test at 95% confidence level). In the remaining 37.5% (6/16) of the cases LM is found to be superior to SVMs. In these runs, SVMs under-perform LMs by about 13.7% on an average. ME on the other hand, performs significantly worse than LM and SVM on all the runs. We also noticed that most of the runs in which LM is superior to SVMs occur on a single collection (disks 1-2) and we intend to perform a data analysis to figure out if the collection has any peculiar characteristics that gives LMs an advantage over SVMs.

Note that the last row of figure 3 reports the best official TREC runs. While the test queries on disk-3 were not used in the ad-hoc task in any of the TRECs, the test set on the WT2G collection consists of only 25 queries. Hence in both cases, we do not have comparable results from TREC. The best runs on the other two datasets had much higher values of average precision than any of our runs. We attribute it to the query expansion techniques used by the participants. In our runs, we focused only on the exact query terms with no expansion.

Our runs show that while MEs are a disappointment, SVMs are more promising for IR. We believe that one can improve on this performance further by including other features such as proximity of query terms, occurrence of query terms as noun-phrases, *etc.* As we have argued in section 3.2, such features would not be easy to incorporate cleanly into the LM framework.

It is interesting to note that while both ME models and SVMs (with linear kernels) are linear discriminative models, SVMs far outperform MEs. We initially suspected that it could be because MEs over-fitted on the training set but we found that the classification accuracy of MEs is much worse than SVMs on the training data too. Hence over-fitting is an unlikely reason for their under-performance. We conjecture that this disparity in performance arises from the different principles governing the two models. MEs learn their discriminant functions by equalizing the model expectations of feature functions with those of the empirical distributions while SVMs try to separate the classes through a maximum margin hyper-plane, minimizing the classification error on training examples. In IR, there are many non-relevant examples that have features similar to those of the relevant ones owing to the inherent ambiguity and complexity of natural language. In this scenario, the differences in feature expectations between the relevant and non-relevant classes may tend to be negligible and hence the ME model fails to discriminate well between the classes. SVMs, on the other hand, try to separate the difficult examples of both classes (called support vectors) using a maximum margin hyper-plane and thus succeed in learning a good discriminant function.

Train ↓ Test →		Disks 1-2 (151-200)	Disk 3 (101-150)	Disks 4-5 (401-450)	WT2G (426-450)
Disks 1-2 (101-150)	LM (μ^* = 1900)	**0.2561 (6.75e-3)**	0.1842	0.2377 (0.80)	0.2665 (0.61)
	SVM	0.2145	0.1877 (0.3)	0.2356	0.2598
	ME	0.1513	0.1240	0.1803	0.1815
Disk 3 (51-100)	LM (μ^* = 500)	**0.2605 (1.08e-4)**	0.1785 (0.11)	0.2503 (0.21)	0.2666
	SVM	0.2064	0.1728	0.2432	0.2750 (0.55)
	ME	0.1599	0.1221	0.1719	0.1706
Disks 4-5 (301-350)	LM (μ^* = 450)	**0.2592 (1.75e-4)**	**0.1773 (7.9e-3)**	**0.2516 (0.036)**	0.2656
	SVM	0.2078	0.1646	0.2355	0.2675 (0.89)
	ME	0.1413	0.0978	0.1403	0.1355
WT2G (401-425)	LM (μ^* = 2400)	**0.2524 (4.6e-3)**	0.1838 (0.08)	0.2335	0.2639
	SVM	0.2199	0.1744	**0.2487 (0.046)**	**0.2798 (0.037)**
	ME	0.1353	0.0969	0.1441	0.1432
Best TREC runs (Site)		0.4226 (UMass)	N/A	0.3207 (Queen's College)	N/A

Figure 3: Comparison of Language models (LM), Support Vector Machines (SVM) and Maximum Entropy model (ME) on ad-hoc retrieval task using title queries: All numbers are in mean average precision. Numbers in brackets indicate the P value from a paired two-tailed T-test. Bold faced numbers indicate that the entry is statistically significant from the nearest run on the same set at 95% confidence level ($\alpha = 0.05$). The value of the optimal Dirichlet smoothing parameter obtained from training the LMs on each collection is given by μ^*.

We observe that language models, despite their inaccurate modeling assumptions are found to be very robust and stable on the ad-hoc retrieval task. While discriminative models tend to be sensitive to noise in the training examples, generative models such as LMs which rely on hand-crafted class-conditional models based on human domain knowledge are relatively impervious to the data-noise and require very little training. This may be considered an important advantage of language models over discriminative models in traditional IR tasks. However, in light of the emergence of modern IR collections such as the web and scientific literature that are characterized by a diverse variety of features, we believe we will increasingly rely on models that can automatically learn these features from a set of labeled examples rather than the hand-crafted models.

We believe that the strength of the SVMs lies precisely in their ability to combine and learn the relative importance of a variety of arbitrary features automatically. To demonstrate this ability, we chose the home-page finding task of TREC-10 where many features such as title, anchor-text and link structure influence relevance. The following subsection presents those experiments in detail.

6.2 Home-page finding on web collection

In the home-page finding task, there are typically only one or two relevant documents per query and the task is to retrieve the relevant document as high in the ranked list as possible. For example the user should be returned the web-page *http://trec.nist.gov* when the query "Text Retrieval Conference" is issued. The corpus used is WT10G, a 10GB collection consisting of web documents. TREC has provided 145 home-page finding queries with relevance judgments and also an additional 100 queries for training [6]. Evaluation is done in terms of three measures:

1. *The mean reciprocal rank (MRR)*: It is the mean of the reciprocal of the rank at which the first relevant document is retrieved across all queries,
2. *Success rate*: the percentage of queries for which an answer is found in the top 10 documents and
3. *Failure rate*: the percent of queries for which no answer is returned in the top 100 documents.

In our experiments, we split the 100 TREC training queries into two sets: the first 50 for training the models and the next 50 for development. We used the 145 home-page queries for testing. We pooled in all the anchor text on links pointed to each document into an anchor-text document. We created three separate indexes: a content index consisting of the textual content of the documents with all the HTML tags removed, an index of the anchor text documents and an index comprising the titles of all documents.

As baselines we used SVMs and language models that use only document-content based features. For LMs, we tuned the Dirichlet smoothing parameter on the 50 training queries and ran the trained model on the development and test queries. Our discriminative model runs consisted of only SVMs this time. As features, we used the 6 features shown in figure 2 from each of the three indexes. We also defined two additional features as defined below:

1. *URL-depth*: It is the reciprocal of the number of branches in the URL-path of a web-document. Home-pages typically are at depth 1, so a depth of unity may indicate relevance.
2. *Link-Factor*: We defined it as $\log(1 + \frac{\text{num-links}(D)}{\text{Avg-num-links}})$ where *num-links(D)* is the number of links pointing to a document D and *Avg-num-links* is its average per document in the collection. Usually popular home-pages have a large number of links pointed to them, hence this feature may add some evidence that a given web-document is a home-page.

In all we have 20 features. As before, we have under-sampled the majority class to solve the unbalanced data problem and used linear kernels once again. To understand the contribution of each type of feature, we built several SVMs trained on different sets of features and tested the performance of each one on the development set. The performance of each of those models on the development set is reported in figure 4.

The results show that the baseline SVM run using only content based features is comparable to the corresponding baseline LM run. As we add anchor-text and title based features to the models, the performance receives a big boost, clearly demonstrating the ability of SVMs to automatically learn various features. The performance has however not been improved by including features of

SVM Features	MRR	Success %	Failure %
Content + Anchor	0.54	73.0	5.2
Content + Anchor + Title	0.61	85.7	10.2
Content + Anchor + Title + URL	0.61	85.7	10.2
Content + Anchor + Title + URL + Link	0.61	85.7	10.2
Language Model baseline	0.35	52.0	10.0
SVM baseline	0.33	53.06	12.24

Figure 4: Comparing the performance contribution of each feature on the development set using SVMs

Model	MRR	Success %	Failure %
full-featured SVM	0.52	77.93	11.03
LM baseline	0.35	57.93	15.86
SVM baseline	0.28	52.41	17.9

Figure 5: Comparison of SVM and LM on the test set

URL-depth and link-factor. Since these two factors do not hurt performance either, we ran the SVM model trained on all the 20 features on our test set. The comparison of performance of full-featured SVM and the baseline runs on the test set is shown in figure 5. The results show that SVMs leverage a variety of features and improve on the baseline LM performance by 48.6% in MRR. It is interesting to note that contrary to our experience, participants of TREC-10 found that features based on link-analysis and URL-depth helped improve the performance substantially. Leveraging these features, the best run in TREC-10 [14] achieved an MRR of 0.77 on the test set. Query-independent features were used in the language modeling approach in the form of prior probabilities while anchor text and content were combined using mixture models. However, their feature weights were optimized using empirical means while our models learn them automatically. Besides, our runs were only preliminary in nature aimed at demonstrating the learning ability of SVMs. We believe there is a lot more that needs to be in defining the right kind of features. For example one could use *PageRank* [22] instead of our link-factor as a feature. Another possibility would be using a two-stage SVM to handle the query-dependent and query-independent features separately. We intend to try out these possibilities in our future work.

7. RELATED WORK

There have been a few attempts in the past in applying discriminative models for IR. In two papers in the early 1980s, Cooper and Huizinga [3] and Cooper [4] make a strong case for applying the maximum entropy approach to the problems of information retrieval. Cooper in particular criticized the absence of any convincing arguments in probabilistic models about term independence assumptions and proposed the maximum entropy approach as a means of getting around such assumptions. In [11], Kantor and Lee extend the analysis of the principle of maximum entropy in the context of information retrieval. Very recently, in 1998, they conducted experiments to test the performance of ME as a method of document retrieval but report discouraging results on large document sets [13]. However, it is worth noting that in all the work mentioned above, the features used in the model were clauses formed by logical connectives of boolean variables representing search terms. Modern IR models, however, have shown that assigning weights to individual terms in a free-text context yields the best retrieval performance. In 2000, Greiff and Ponte [8] showed that the classic binary independence model and the maximum entropy approach are equivalent. However, their work did not consist of any experiments with the maximum entropy model.

In two papers [5, 7] that can be called the closest to the current work, the authors suggested the method of logistic regression, which is equivalent to the method of maximum entropy we used in our work. Experiments reported in [7] show that the performance of logistic regression is statistically indistinguishable from that of the vector space model. They also achieved statistically significant improvement over the vector space model on one of the collections when the feature values are normalized by their sample means and standard deviations. However, the collections used in their experiments were quite small (of the order of 2000 documents each) compared to the modern day IR collections (order of 1,000,000 documents) on which current systems are evaluated.

One fundamental modeling issue that distinguishes the present work from that of [5, 7] is that in the latter, each training example is built from feature vectors defined on each query term and document pairs. For example, let us say query Q consists of the terms q_1 and q_2 and document D that is relevant to the query has term specific features $[f_1(q_1, D), f_2(q_1, D)]$ corresponding to q_1 and $[f_1(q_2, D), f_2(q_2, D)]$ corresponding to q_2. Then for the query-document pair (Q, D), the model of [5, 7] produces two unique training examples given by the vectors $[f_1(q_1, D), f_2(q_1, D), R = 1]$ and $[f_1(q_2, D), f_2(q_2, D), R = 1]$. In contrast, our model generates a single training example for each query-document pair given by $[\sum_{i=1}^{2} f_1(q_i, D), \sum_{i=1}^{2} f_2(q_i, D), R = 1]$. Now, it is sometimes possible that relevance of the document D to the query Q is entirely influenced by one of the query terms q_2. (For example, $[q_1 = the, q_2 = Beatles]$). In this scenario, the training algorithm of [5, 7] is misled into believing that the example based on features of q_1 is relevant, although we know its contribution to relevance is negligible. On the other hand, defining features cumulatively over all the query terms ensures higher consistency among training examples.

8. CONCLUSIONS AND FUTURE WORK

In this work, we have argued in favor of the view that IR should be treated as a problem of binary classification of relevance as this not only helps quantify relevance explicitly, but also permits us apply sophisticated pattern classification techniques to IR. We then presented the popular IR probabilistic models from the perspective of pattern classification and showed that they are generative in nature. We explored the applicability of discriminative classifiers such as SVMs and MEs to IR. We argued that besides their attractive theoretical properties, their main utility to IR lies in their ability to learn automatically a variety of features that influence relevance. Our experiments on ad-hoc retrieval show that using the same type of features, SVMs perform as well as LMs on most of our runs. We also demonstrate the ability of SVMs in learning a variety of features in our home-page finding task, where they outperform the baseline runs that use only content-based features by about 50% in MRR.

Our results are intended to be only demonstrative in nature and there is a lot of room for further improvement through better feature engineering and by leveraging a huge body of literature on SVMs and other sophisticated pattern classification algorithms. We believe that the most important contributions of this work are our argument in favor of applying discriminative classifiers to IR and our demonstration of the potential of SVMs in learning arbitrary features automatically. We envision that further study of these tech-

niques would only result in advances in IR systems towards better capturing the users' information needs.

As part of our future work, we plan to evaluate the performance of SVMs on ad-hoc retrieval task with longer queries and enhanced features such as proximity of query terms, synonyms, *etc*. We also intend to study user modeling by incorporating user-preferences as features in the SVM.

Acknowledgments

The author would like to thank James Allan, Victor Lavrenko and the anonymous reviewers for their useful feedback and comments. This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSC EN-SD grant number N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the author's and do not necessarily reflect those of the sponsor.

9. REFERENCES

[1] Berger, A. L., Della Pietra, D., Stephen A. and Della Pietra, V. J., A Maximum Entropy Approach to Natural Language Processing, *Computational Linguistics*, vol. 22(1), p39-71, 1996.

[2] Burges, C., A Tutorial on Support Vector Machines for Pattern Recognition, *Data Mining and Knowledge Discovery*, vol. 2(2), p121-167, 1998.

[3] Cooper, W. S. and Huizinga, P., The maximum entropy principle and its application to the design of probabilistic retrieval systems, *Information Technology, Research and Development*, 1:99-112, 1982.

[4] Cooper, W. S., Exploiting the maximum entropy principle to increase retrieval effectiveness, *Journal of the American Society for Information Science*, 34(1):31-39, 1983.

[5] Cooper, W. S., Gey, F. and Dabney, D., Probabilistic Retrieval based on Staged Logistic regression, *ACM SIGIR*, p198-210, 1992.

[6] Craswell, N., Home-page finding training queries, http://es.cmis.csiro.au/TRECWeb/Qrels/homepages.wt10g.training01.

[7] Gey, F., Inferring probability of relevance using the method of logistic regression, *ACM SIGIR*, p222-231, 1994.

[8] Greiff, W. R. and Ponte, J. M., The maximum entropy approach and probabilistic IR models, *ACM Trans. on Information Systems*, 18(3):246-287, 2000.

[9] Harter, S.P., A probabilistic approach to automatic keyword indexing. Part I: On the distribution of specialty words in a technical literature, *Journal of the ASIS*, vol. 26, 197-206.

[10] Hawking, D. and Craswell, N., Overview of the TREC-2001 web track, *TREC proceedings*, 2001.

[11] Kantor P. B. and Lee, J. J., The maximum entropy principle in information retrieval, SIGIR, 1986.

[12] Joachims, T., Text categorization with support vector machines: learning with many relevant features, Proceedings of *10th European Conference on Machine Learning*, p137-142, 1998.

[13] Kantor P. B. and Lee, J. J., Testing the maximum entropy principle for information retrieval, *Journal of the American Society for Information Science*, 49(6):557-566, 1998.

[14] Kraaij, W., Westerveld T. and Hiemstra, D., The importance of prior probabilities for entry page search, *SIGIR*, pages 27-34, 2002.

[15] Lafferty, J. and Zhai, C., Probabilistic relevance models based on document and query generation, Workshop on *Language Modeling and Information Retrieval*, 2001

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

[17] Malouf, R., A comparison of algorithms for maximum entropy parameter estimation, http://citeseer.nj.nec.com/malouf02comparison.html.

[18] Nallapati, R. and Allan, J., Capturing Term Dependencies using a Sentence Tree based Language Model, *CIKM*, 2002.

[19] Ng., A. and Jordan, M., On Discriminative vs. Generative classifiers: A comparison of logistic regression and naïve Bayes, *Neural Information Processing Systems*, 2002.

[20] Nigam, K., Lafferty, J. and McCallum, A., Using maximum entropy for text classification, *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61-67, 1999.

[21] Ogilvie, P., and Callan J., Combining Document Representations for Known Item Search, *SIGIR*, 2003.

[22] Page, L., Brin, S., Motwani, R. and Winograd, T., The PageRank Citation Ranking: Bringing Order to the Web, *Stanford Digital Library Technologies Project*, 1998.

[23] Ponte, J. M. and Croft, W. B., A Language Modeling Approach to Information Retrieval, *ACM SIGIR*, 275-281, 1998.

[24] Ratnaparkhi, A., A Maximum Entropy Part-Of-Speech Tagger, *Empirical Methods in Natural Language Processing*, 1996.

[25] Robertson S. E. and Sparck Jones, K., Relevance weighting of search terms, *Journal of American Society for Information Sciences*, 27(3):129-146, 1976.

[26] Robertson, S. E., On Bayesian models and event spaces in information retrieval, Workshop on *Mathematical and Formal methods for IR*, 2002.

[27] Robertson, S. E., van Rijsbergen, C.J., and Porter, M. F., Probabilistic models of indexing and searching, *Proceedings of SIGIR*, 1980.

[28] Salton, G., The SMART Retrieval System - Experiments in Automatic Document Processing, *Prentice hall Inc.*, Englewood Cliffs, NJ, 1971.

[29] Teevan, J. and Karger, D., Empirical Development of an Exponential Probabilistic Model for Text Retrieval: Using Textual Analysis to Build a Better Model, In Proceedings of the 26th Annual ACM Conference on Research and Development in Information Retrieval, 2003.

[30] Vapnik, V. N., Statistical Learning Theory, *John Wiley & Sons*, 1998.

[31] Zhai, C. and Lafferty, J., A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval, *SIGIR*, 2001.

[32] Zhang, J. and Mani, I., kNN approach to unbalanced data distributions: A case study involving Information Extraction, *Workshop on learning from imbalanced datasets II*, ICML, 2003.

[33] Zhang, L., A Maximum Entropy Modeling Toolkit for Python and C++, http://www.nlplab.cn/zhangle/maxent.html.

[34] Language Modeling Toolkit for Information Retrieval, http://www-2.cs.cmu.edu/ lemur/.