A Weighted Maximum Entropy Language Model for Text Classification

Kostas Fragos[1], Yannis Maistros[2], Christos Skourlas[3]

1 Department of Computer Engineering, National technical University of Athens, Iroon Polytexneiou 9 15780 Zografou Athens Greece
http://nts.ece.ntua.gr/nlp_lab
kfragos@ece.ntua.gr
2 Department of Computer Engineering, National technical University of Athens, Iroon Polytexneiou 9 15780 Zografou Athens Greece
maistros@ece.ntua.gr
3 Department of Computer Science, Technical Educational Institute of Athens, Ag Spyridonos 12210 Aigaleo Athens Greece
cskourlas@teiath.gr

Abstract. The Maximum entropy (ME) approach has been extensively used for various natural language processing tasks, such as language modeling, part-of-speech tagging, text segmentation and text classification. Previous work in text classification has been done using maximum entropy modeling with binary-valued features or counts of feature words. In this work, we present a method to apply Maximum Entropy modeling for text classification in a different way it has been used so far, using weights for both to select the features of the model and to emphasize the importance of each one of them in the classification task. Using the X square test to assess the contribution of each candidate feature from the obtained X square values we rank the features and the most prevalent of them, those which are ranked with the higher X square scores, they are used as the selected features of the model. Instead of using Maximum Entropy modeling in the classical way, we use the X square values to weight the features of the model and give thus a different importance to each one of them. The method has been evaluated on Reuters-21578 dataset for test classification tasks, giving very promising results and performing comparable to some of the "state of the art" systems in the classification field.

1. Introduction

With the volume of electronic digital documents increasing rapidly today, there is a significant interest in developing tools and techniques that help people to better organize and manage these resources. Human categorization is very time-consuming and costly and thus its applicability is limited especially for very large document collections. Consequently, text classification techniques have increased in importance and economic value for digital world as they develop key technologies for classifying new electronic documents, finding interesting information on web and guiding a user's search through hypertext.
In early approaches to text classification a document representation model was employed, usually in a term-based vector in some high dimensional Euclidean space where each dimension corresponds to a term, with some classification algorithm, trained in a supervised learning manner. Up to now, a great many of text categorization and classifying techniques have been proposed to the literature, including Bayesian techniques [1],[2],[3], *k*-nearest neighbors (*k*NN) classification methods [4],[5],[6], the so-called Rocchio algorithm from information retrieval [7],[8], artificial neural networks (ANN) techniques [9],[10],[11],[12], support vector machines (SVM) learning method [13],[14],[15], hidden Markov models (HMM) [19],[20], and decision tree (DT) classification methods [17],[18],[9],[1]. In most of these methods, the aim is to estimate the parameters of the joint distribution between the object X being classified and a class category C and assign the object to that category with the greater probability. Unfortunately, in most real-world applications the joint distribution is usually unavailable due to the complexity of the problem. In general it cannot be computed efficiently since it would involve calculations over all possible combinations of X, C, a potentially infinite set. Instead, using the Bayes formula the problem can be

decomposed to the estimation of two components *P(X|C)* and *P(C)*, known as the conditional class distribution and prior distribution, respectively.

Maximum Entropy (ME) modeling is a general and intuitive way for estimating a probability from data and it has been successfully applied in various natural language processing tasks such as language modeling, part-of-speech tagging and text segmentation [23],[24],[25],[26],[28],[29]. The principle underlying ME is that the estimated conditional probability should be as uniform as possible, that is, have the maximum entropy. The main advantage of ME modeling for the classification task is that offers a framework for specifying any arbitrary relevant information we believe it might contribute to the classification task. This relevant information is expressed in the form of feature functions, the mathematical expectations (constraints) of which are estimated upon labeled training data and characterize the class-specific expectations for the distribution. The principle of ME is clear: among all the allowed probability distributions which conform to the constraints of the training data chose the one with the maximum entropy, that is, the most uniform. It can be proved that there is a unique solution for this problem. The uniformity of the found solution, known as the "lack of smoothing", may be undesirable to some cases, for example, if we have a feature that always predict a certain class, then this feature may get an excessively high weight. Another shortcoming of the ME modeling is that the algorithm which is used to find the solution can be computationally expensive due to complexity of the problem.

In this work, we try to eliminate the above undesirable situations. As it is well known, *X* square statistic has been widely used in natural language processing tasks. The *X* square test for independence can be applied to problems where the data is divided into mutual exclusive categories and has the advantage, unlike the other tests that it does not assume normally distributed probabilities. The essence of the test is to assess the assumption about the independence of a data object with a category comparing the observed frequency of that object with the category and the expected frequency for independence. If the difference between the observed and expected frequency is large, then we can reject the assumption about independence (null hypothesis). In our case, if we think every *word* term *w* in a document *d* as a candidate feature we can use the *X* square statistic to test the independence of this *word* with each one of the class categories *c*, simply by counting the observed frequencies of the word in each class category in the training set. The resulting value of the test is then used to select the most representative features for the maximum entropy model as well as to weight the features giving different importance in the classification task in each one of them.

In what it follows we present, in section 2 the application of the *X* square test in our data for feature selection and the weighting scheme, in section 3 the maximum entropy modeling and the improved iterative scaling (IIS) algorithm, in section 4 we discuss the way of using maximum entropy modeling for text classification, in section 5 experimental results are given and finally in section 6 we conclude with a discussion about our method and the similar works.

2. X Square Test for Feature Selection

Among the most challenging tasks in the classification process, we can distinguish the selection of suitable features to represent the instances of a particular class. Additively, selection of the best candidate features can be a real disadvantage for the selection algorithm, in both means of effort and time consumption [22].
As we have mentioned above, each document is represented as a vector of words, as is typically done in information retrieval. Although in most text retrieval applications, the entries in the vector are weighted to reflect the importance of the term in retrieval, in text classification simpler binary feature values (i.e., a term either occurs or does not occur in a document) are often used instead. Usually, text collections contain millions of unique terms and for reasons of computational efficiency and efficacy, feature selection is an essential step when applying machine learning methods to text categorization. In this work, the *X* square test is used to reduce the dimensionality of data and for the weighting purposes of the maximum entropy modeling.

In 1900, Karl Pearson developed a statistic that compares all the observed and expected numbers when the possible outcomes are divided into mutually exclusive categories. The form in eq.1 gives the chi-square statistic:

$$X^2 = \sum \frac{(observed - expected)^2}{expected} \quad (1)$$

Where the Greek letter **Σ** stands for summation and is calculated over the categories of possible outcomes.

The *observed* and *expected* values can be explained in the context of hypothesis testing. If we have data that are divided into mutual exclusive categories and form a null hypothesis about that data, then the expected value is the value of each category if the null hypothesis is true. The *observed* value is the value for each category that we observe from the sample data.

The chi-square test is a remarkably versatile way of gauging the significance of how closely the data agree with the detailed implications of a null hypothesis.

To clarify things let us see an example with real data from Reuters-21578 dataset and specifically tailored for the classification task. Suppose we have two distinct class categories c1='Acq' and c2≠'Acq' from the Reuters-21578 'ModApte' split training dataset and we are interested in assessing the independence of the word 'usa' with the class categories c1 and c2. From this training dataset we removed all numbers and the words of a stopword list. Counting the frequencies of the word 'usa' in the training dataset we find that the word 'usa' appears with class *Acq* (c1='Acq') 1,238 times, with the other classes, that is not the class 'Acq' (c2≠'Acq') 4,464 times. In the class 'Acq' there is a total of 125,907 word terms while in the other classes a total of 664,241. This is equivalent to a total of *N*=790,148 word terms overall in the Reuters-21578 training dataset. It would be useful to use the contingency table 1 in which the data are classified.

Table 1. Contigency table of frequencies for the word *usa* and the class *Acq* from the Reuters-21578 'ModApte' split training dataset

	c_1 = 'Acq'	$c_2 \neq$ 'Acq'	Total
w = *'usa'*	1,238	4,464	**5,702**
w ≠ 'usa'	124,669	659,777	**784,446**
Total	**125,907**	**664,241**	**N=790,148**

Moreover, using maximum likelihood estimates we can compute the probabilities of class 'Acq' and the word 'usa' as follows.

P(c1='Acq') = 125,907/790,148=0.1593

P(w='usa') = 5,702/790,148=0.0072

The assumption about the independence (null hypothesis) is that occurrences of the word 'usa' and the class label 'Acq' are independent. We compute now the probability of the null hypothesis.

H_0 : *P*('usa','Acq') = *P*('usa') x *P*('Acq') = 0.0072 x 0.1593 = 0.0011

Then we calculate the X^2 value using eq. 1. Looking up the X^2 distribution from tables or by using appropriate statistical software, we find a critical value for a significance level *a* (usually *a=0.05*) and for one degree of freedom (the statistic has one degree of freedom for a 2x2 contingency table). If the calculated value is greater than the critical value we can reject the null hypothesis that the word 'usa' and the class label 'Acq' occur independently. So, for a large calculated X^2 value we have a strong evidence for the pair ('usa', 'Acq'). The word 'usa' is then a good feature for the classification in the category 'Acq'.

To make simpler the things, we are only interested in large calculated X^2 values and not to reject the null hypothesis. Our aim is to select the most representative features among the large number of candidates and perform classification in a lower dimensionality space.

We give now a simpler form that we use in this paper for the calculation of X^2 values. For a contingency 2-by-2 table, the *X* square values can be calculated by the following form:

$$X^2 = \frac{N(a_{11}a_{22}-a_{12}a_{21})^2}{(a_{11}+a_{12})(a_{11}+a_{21})(a_{12}+a_{22})(a_{21}+a_{22})} \quad (2)$$

Where a_{ij} are the entries of the contingency 2-by-2 table A and N the total sum of these entries.

4. Maximum Entropy Approach

3.1 Maximum Entropy Modeling

Entropy has its original back in the dates of Shannon [27] when it was originally used to estimate how much of the data can be compressed before they are transmitted over a communication channel. The entropy H itself measures the average uncertainty of a single random variable X:

$$H(p) = H(X) = \sum_{x \in X} p(x) \log_2 p(x) \quad (3)$$

Where, $p(x)$ is the probability mass function of the random variable X. The eq. 4 tells us the average bits we need to transfer all the information in X.

In its use in the communication theory to save the bandwidth of a communication channel, we prefer a model of X with less entropy so that we can use smaller bits to interpret the uncertainty (information) inside X. However, in its use in natural language processing tasks, we want to find a model to maximize the entropy. This sounds as though we are violating the basic principle in entropy. Actually, the main reason to do so is to preserve as less bias as possible when the certainty cannot be identified from the empirical evidence.

Many problems in natural language processing can be re-formulated as statistical classification problems. Specifically, in text classification we think the text classification task to be a random process Y which takes as input a document d and produces as output a class label c. The output of the random Y may be affected by some contextual information X, whose domain is all the possible textual information contained in the document d. Our aim is to specify a model $p(y|x)$ which denotes the probability that the model assigns to $y \in Y$ when the contextual information is $x \in X$.

At the first step, we observe the behavior of the random process in a training sample set collecting a large number of samples $(x_1, y_1), (x_2, y_2) \ldots (x_N, y_N)$. We can summarize the training sample defining a joint empirical distribution over x and y from these samples:

$$\tilde{p}(x,y) = \frac{1}{N} \times number\ of\ times\ (x,y)\ occurs\ in\ the\ sample \quad (4)$$

One way to represent contextual evidence is to encode useful facts as features and to impose constraints on values of those feature expectations. This is done by the following way. We introduce the indicator function

$$f(x,y) = \begin{cases} 1 & if\ y='some\ particular\ value\_1'\ \ and\ \ x='some\ particular\ value\_2' \\ 0 & otherwise \end{cases} \quad (5)$$

For example, in our classification problem an indicator function may be $f(x,y)=1$ if $y='c_1'$ and x contains the word *'money'* and $f(x,y)=0$ otherwise. Where $'c_1'$ is a particular value from the class labels and x is the context (the document) where the word *'crude'* occurs within. Such an indicator function f is called feature function or feature for short. Its mathematical expectation with respect to the model $p(y|x)$ is

$$\sum_{x,y} \tilde{p}(x,y) f(x,y) \quad (6)$$

We can acknowledge the importance of this statistic by requiring that the expected value, the model assigns to the corresponding feature function is in accordance with the empirical expectation of eq. 7.

$$\sum_{x,y} \tilde{p}(x)\, p(y \mid x) f(x,y) = \sum_{x,y} \tilde{p}(x,y) f(x,y) \quad (7)$$

where $\tilde{p}(x)$ is the empirical distribution of x in the training sample.

We call the requirement eq. 8 a *constraint equation* or simply a *constraint*.

When constraints are estimated in this fashion, there are many conditional probability models which can satisfy the constraints. Among all these models there is always a unique distribution that has the maximum entropy and it can be shown [] that the distribution has an exponential form:

$$p_\lambda(y \mid x) = \frac{1}{Z_\lambda(x)} \exp\left(\sum_i \lambda_i f_i(x,y) \right) \quad (8)$$

where *Z(x)* a normalizing factor to ensure a probability distriution given by

$$Z_\lambda(x) = \sum_y \exp\left(\sum_i \lambda_i f_i(x,y) \right) \quad (9)$$

where λ_i a parameter associated with the constraint f_i to be estimated.

The solution to maximum entropy model in the form of eq. 9 is also the solution to a dual maximum likelihood problem for models of the same exponential form. It is guaranteed that the likelihood surface is convex, having a single global maximum and no local maxima and there is an algorithm that finds the solution performing hillclimbing in likelihood space.

3.2 Improved Iterative Scaling

We describe now a basic outline of the improved iterative scaling (IIS) algorithm, a hillclimbing algorithm for estimating the parameters λ_i of the maximum entropy model, specially adjusted for text classification. The notation of this section follows that of Nigam et al. [31] with x to represent a document d and y a class label c.

Given a set of training dataset D, which consists of pairs *(d, c(d))*, where d the document and *c(d)* the class label in which the document belongs, we can calculated the loglikelihood of the model of eq. 9.

$$L(p_\lambda \mid D) = \log \prod_{d \in D} p_\lambda(c(d) \mid d) = \sum_{d \in D} \sum_i \lambda_i f_i(d, c(d)) - \sum_{d \in D} \log \sum_c \exp \sum_i \lambda_i f_i(d,c) \quad (10)$$

The algorithm is applicable whenever the feature functions $f_i(d,c(d))$ are non-negative.

To find the global maximum of the likelihood surface, the algorithm must start from an initial exponential distribution of the correct form that is to guess a starting point and then perform hillclimbing in likelihood space. So, we start from an initial value for the parameters λ_i , say $\lambda_i = 0$ for $i=1:K$ (where K the total number of features) and in each step we improve by setting them equal to $\lambda_i + \delta i$, where δi the increment quantity. It can be shown that at each step we can find the best δi by solving the equation:

$$\sum_{d \in D} (f_i(d, c(d)) = \sum_c p_\lambda(c \mid d) \exp(\delta_i f^{\#}(d,c)) \quad (11)$$

Where $f^{\#}(d,\mathrm{c})$ is the sum of all features in training instance d.

Equation 12 can be solved in a closed form if the is $f^{\#}(d,\mathrm{c})$ is constant, say M, for all *d, c*.

$$\delta_i = \frac{1}{M} \log \frac{\sum_{d \in D} f_i(d, c(d))}{\sum_c p_\lambda(c \mid d) f_i(c, d)} \quad (12)$$

where $p_\lambda(c|d)$ is the distribution of the exponential model of eq. 9.

If this is not true, then eq. 12 can be solved with a numeric root-finding procedure, such as Newton's method.

However in the last case, we can still solve eq. 12 in closed form by adding an extra feature to provide $f^{\#}(d,c)$ to be constant for all d, c in the following way:

we define M as the greatest possible feature sum:

$$M = \max_{d,c} \sum_{i=1}^{K} f_i(d,c) \quad (13)$$

and add an extra feature, that is defined as follows:

$$f_{K+1}(d,c) = M - \sum_{i=1}^{K} f_i(d,c) \quad (14)$$

Now we have all the pieces to summarize the improved iterative scaling algorithm (IIS)

```
Begin
  Add an extra feature f_{K+1} following eq. 14,15
  Initialize λ_i =0 for i=1:K+1
  Repeat
    ●Calculate the expected class labels p_λ(c|d) for each
    document with the current parameters using eq.9
    ●calculate δ_i from eq. 13
    ●set λ_i= λ_i+ δ_i
  Until convergence
  Output: Optimal parameters λ_i optimal model p_λ
End
```

4. Maximum Entropy Modeling for Text Classification

The basic shortcoming of the IIS algorithm is that may be computationally expensive due to complexity of the classification problem. Moreover, the uniformity of the found solution (lack of smoothing) can also cause problems. For example, if we have a feature that always predict a certain class, then this feature may get an excessively high weight. Our innovation in this work is to use the X square test to rank all the candidate feature words, that is, all the word terms that appear in the training set and then select the most high ranked of them for using in the maximum entropy model.

If we decide to select the K most high ranked word terms $w_1, w_2, \ldots, w_K$ we instantiate the features as follows:

$$f_i(d,c) = \begin{cases} xsquare(i) & if\ the\ word\ \ w_i\ \ occurs\ in\ d\ \ and\ the\ pair\ (d,c)\ \ appears\ in\ the\ training\ set \\ 0 & otherwise \end{cases} \quad \textbf{(15)}$$

where *xsquare(i)* denotes the *X square* score of the word w_i obtained during the feature selection phase. This way of instantiating features has two advantages: first it gives a weight to each feature and second it creates a separate list of features for each class label. These features are activated only with the presence of the particular class label and are strong indicators of it. Of course some features participate to more than one lists, that is, are common to more than one classes. These lists of features are used from the resulting binary text classifier (the optimal model of the IIS algorithm) to calculate the expected class labels probabilities for a document *d*, eq 9, and then to assign the document *d* to the class with the higher probability.

5. Experimental Results

We evaluated our method using the "ModApte" split of the Reuters-21578 dataset compiled by David Lewis. The "ModApte" split leads to a corpus of 9,603 training documents and 3,299 test documents. Of the 135 potential topic categories we choose to evaluate only over 10 categories for which there is enough number of training and test document examples. Because we want to built a binary classifier we split the documents into 2 groups: 'Yes' group, the document belongs to the category and 'No' group, the document do not belong to the category. The 10 categories with the number of documents for the training and test phase are shown in table 1.

Table 2. 10 categories from the "ModApte" split of the Reuters-21578 dataset with the number of documents for the Training and the Test phase for a binary classifier.

	Train		Test	
Category	Yes	No	Yes	No
Acq	1615	7988	719	2580
Corn	175	9428	56	3243
Crude	383	9220	189	3110
Earn	2817	6786	1087	2212
Grain	422	9181	149	3150
Interest	343	9260	131	3168
Money-fx	518	9085	179	3120
Ship	187	9416	89	3210
Trade	356	9247	117	3182
Wheat	206	9397	71	3228

In the training phase we parsed all 9,603 documents. We did not stem the words, simply we removed all numbers and the words from a stopword list. After this preprocessing phase we ended up with 32,412 discrete terms of a total of 790,148 word terms. The same preprocessing phase was followed in the test phase.

We applied the *X* square test on the corpus of those features as exactly is described in the section 3 and then we selected for the maximum entropy model the most 2,000 higher ranked word terms for each category. Table 3 presents for each category the 10 top ranked word terms by the *X* square test

Table 3. 10 top ranked words by the X square test for the 10 categories from the ModApte Reuters-21578 training dataset

Acq	**Corn**	**Crude**	**Earn**	**Grain**	**Interest**	**Money-fx**	**Ship**	**Trade**	**Wheat**
bgas	values	crude	earn	filing	money	flexible	acq	trade	rumors
annou	july	comment	usa	prevailing	fx	conn	deficit	brazil	monetary
ameritech	egypt	spoke	convertible	outlined	discontinued	proposals	buy	agreement	eastern
calny	agreed	stabilizing	moody	brian	africa	soon	officials	chirac	policy
adebayo	shipment	cancel	produce	marginal	signals	requirement	price	communications	cbt

echoes	belgium	shipowners	former	winds	anz	slow	attempt	growth	storage
affandi	oilseeds	foresee	borrowings	proceedings	exploration	soybeans	mitsubishi	restraint	proposal
f8846	finding	sites	caesars	neutral	program	robert	mths	ran	reuter
faded	february	techniques	widespread	requiring	tuesday	calculating	troubled	slowly	usually
faultered	permitted	stayed	honduras	bangladesh	counterparty	speculators	departments	conclusion	moisture

The 2000 higher ranked word terms from each category are then used to instantiate the features of the maximum entropy model (WMEM) as exactly it was described in section 4. Using a number of 200 iterations in the training phase of classifier, the IIS algorithms outputs the optimal $\lambda i's$, that is the optimal model $p_{\lambda}(c|d)$. We call this method weighted maximum entropy modeling, to emphasize the event that we use selected features and give to them a weight.

Table 4. Micro-average Breakeven performance for 5 different learning algorithms explored by Dumais et al.

	Findsim	NBayes	BayesNets	Trees	LinearSVM
earn	92.9%	95.9%	95.8%	97.8%	98.2%
acq	64.7%	87.8%	88.3%	89.7%	92.7%
money-fx	46.7%	56.6%	58.8%	66.2%	73.9%
grain	67.5%	78.8%	81.4%	85.0%	94.2%
crude	70.1%	79.5%	79.6%	85.0%	88.3%
trade	65.1%	63.9%	69.0%	72.5%	73.5%
interest	63.4%	64.9%	71.3%	67.1%	75.8%
ship	49.2%	85.4%	84.4%	74.2%	78.0%
wheat	68.9%	69.7%	82.7%	92.5%	89.7%
corn	48.2%	65.3%	76.4%	91.8%	91.1%
Avg Top 10	64.6%	81.5%	85.0%	88.4%	91.3%
Avg All Cat	61.7%	75.2%	80.0%	N/A	85.5%

To evaluate the classification performance of the binary classifiers we use the so-called *precision/recall breakeven point*, which is the standard measure of performance in text classification and it is defined as the value for which *precision* and *recall* are equal. *Precision* is the proportion of items placed in the category that are really in the category, and Recall is the proportion of items in the category that are actually placed in the category. Table 3 summarizes the breakeven point performance for 5 different learning algorithms explored by Dumais et al. [32] for the 10 most frequent categories as well as the overall score for all 118 categories.

Table 5. Breakeven performance of the weighted maximum entropy model over the top 10 categories of the Reuters-21578 dataset

Weighted Maximum Entropy Model (WMEM) performance	
Category	Breakeven point
Acq	87.93%
Corn	57.36%
Crude	72.20%
Earn	97.98%
Grain	83.37%
Interest	64.21%
Money-fx	75.09%
Ship	50.22%
Trade	48.16%
Wheat	69.88%

The results in table 5 show that our method performs well the larger categories. It performs better than the other classifiers in the 'money-fx' category and outperforms most of the other classifiers in some of the largest in testing size categories like 'earn', 'acq' and 'grain'.

6. Discussion and Similar Work

To our knowledge at least three other works have used maximum entropy for text classification: The work of Ratnaparkhi, a very preliminary experiment that uses binary features. The work of Mikheev [33] examined the performance of maximum entropy modeling and feature selection for text classification on the RAPRA corpus, a corpus of technical abstracts. Again in this work binary features were used. Nigam et al. [31] used counts of occurrences instead of binary features and the showed that maximum entropy is competitive with and sometimes better than naïve Bayes classifier.
In this work, we extended these previous works both, using a feature selection strategy and assigning weights to the features with the *X square* test. The results of the evaluation are very promising. However, it is needed, to further continue the experiments at least to two directions: first, to perform experiments changing the number of the selected features or the selection strategy, as well as, the number of the iterations in the training phase and second, to perform additional experiments using alternative datasets such as, the *WebKB* dataset, the *Newsgroup*s dataset etc., in order to have a better idea about the performance of the method. These remain for a future work.

Acknowledgements

This work was co-funded by 75% from E.U. and 25% from the Greek Government under the framework of the Education and Initial Vocational Training Program – Archimedes.

6. References

1. Lewis, D. and Ringuette, M., A comparison of two learning algorithms for text categorization. In The Third Annual Symposium on Document Analysis and Information Retrieval, pp.81-93, 1994.
2. Makoto, I. and Takenobu, T., Cluster-based text categorization: a comparison of category search strategies, In ACM SIGIR'95, pp.273-280, 1995.
3. McCallum, A. and Nigam, K., A comparison of event models for naïve Bayes text classification, In AAAI-98 Workshop on Learning for Text Categorization, pp.41-48, 1998.
4. Masand, B., Lino, G. and Waltz, D., Classifying news stories using memory based reasoning, In ACM SIGIR'92, pp.59-65, 1992.
5. Yang, Y. and Liu, X., A re-examination of text categorization methods, In ACM SIGIR'99, pp.42-49, 1999.
6. Yang, Y., Expert network: Effective and efficient learning from human decisions in text categorization and retrieval, In ACM SIGIR'94, pp.13-22, 1994.
7. Buckley, C., Salton, G. and Allan, J., The effect of adding relevance information in a relevance feedback environment, In ACM SIGIR'94, pp.292-300, 1994.
8. Joachims, T., A probabilistic analysis of the rocchio algorithm with TFIDF for text categorization, In ICML'97, pp.143-151, 1997.
9. Guo, H. and Gelfand S. B., Classification trees with neural network feature extraction, In IEEE Trans. on Neural Networks, Vol. 3, No. 6, pp.923-933, Nov., 1992.
10. Liu, J. M. and Chua, T. S., Building semantic perceptron net for topic spotting, In ACL'01, pp.370-377, 2001.
11. Ruiz, M. E. and Srinivasan, P., Hierarchical neural networks for text categorization, In ACM SIGIR'99, pp.81-82, 1999.
12. Schutze, H., Hull, D. A. and Pedersen, J. O., A comparison of classifier and document representations for the routing problem, In ACM SIGIR'95, pp.229-237, 1995.
13. Cortes, C. and Vapnik, V., Support vector networks, In Machine Learning, Vol.20, pp.273-297, 1995.
14. Joachims, T., Learning to classify text using Support Vector Machines, Kluwer Academic Publishers, 2002.
15. Joachims, T., Text categorization with Support Vector Machines: learning with many relevant features, In ECML'98, pp.137-142, 1998.
16. Schapire, R. and Singer, Y., BoosTexter: A boosting-based system for text categorization, In Machine Learning, Vol.39, No.2-3, pp.135-168, 2000.
17. Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C.J., Classification and Regression Trees, Wadsworth Int. 1984.

18. Brodley, C. E. and Utgoff, P. E., Multivariate decision trees, In Machine Learning, Vol.19, No.1, pp.45-77, 1995.
19. Denoyer, L., Zaragoza, H. and Gallinari, P., HMM-based passage models for document classification and ranking, In ECIR'01, 2001.
20. Miller, D. R. H., Leek, T. and Schwartz, R. M., A Hidden Markov model information retrieval system, In ACM SIGIR'99, pp.214-221, 1999.
21. Kira, K. and Rendell, L. A practical approach to feature selection. In Proc. 9th International workshop on machine learning (pp. 249-256) 1992.
22. Gilad-Bachrach, Navot A., Tishby N. Margin Based Feature Selection - Theory and Algorithms. In Proc of ICML 2004
23. Stanley F. Chen and Rosenfeld R. A Gaussian prior for smoothing maximum entropy models. Technical report CMU-CS-99108, Carnegie Mellon University, 1999.
24. Ronald Rosenfeld. Adaptive statistical language modelling: A maximum entropy approach, PhD thesis, Carnegie Mellon University, 1994.
25. Ratnparkhi Adwait, J. Reynar, S. Roukos. A maximum entropy model for prepositional phrase attachment. In proceedings of the ARPA Human Language Technology Workshop, pages 250-255, 1994.
26. Ratnparkhi Adwait. A maximum entropy model for part-of-speech tagging. In Proceedings of the Empirical Methods in Natural Language Conference, 1996.
27. Shannon C.E. 1948. *A mathematical theory of communication*. Bell System Technical Journal 27:379 – 423, 623 – 656.
28 Berger A. 1996. *A Brief Maxent Tutorial*. http://www-2.cs.cmu.edu/~aberger/maxent.html.
29 Berger A. 1997. *The improved iterative scaling algorithm: a gentle introduction*. http://www-2.cs.cmu.edu/~aberger/maxent.html
30 Della Pietra S., V. Della Pietra and J. Lafferty. Inducing features of random fields. IEEE transaction on Pattern Analysis and Machine Intelligence, 19(4), 1997.
31 Nigam K., J. Lafferty, A. McCallum. Using maximum entropy for text classification, 1999.
32 Dumais, S. T., Platt, J., Heckerman, D., and Sahami, M. Inductive learning algorithms and representations for text categorization. *Submitted for publication*, 1998. http://research.microsoft.com/~sdumais/cikm98.doc
33 Mikheev A. Feature Lattics and maximum entropy models. In *machine Learning*, McGraw-Hill, New York, 1999.