On Text-Based Estimation of Document Relevance

Eerika Savia[1] and Samuel Kaski[1,2]
[1]Neural Networks Research Centre
Helsinki University of Technology
P.O.Box 5400, FIN-02015 HUT, Finland
[2]Department of Computer Science
P.O.Box 26, FIN-00014 University of Helsinki,Finland
E-mail: {Eerika.Savia,Samuel.Kaski}@hut.fi

Ville Tuulos and Petri Myllymäki
Complex Systems Computation Group
Helsinki Institute for Information Technology
University of Helsinki & Helsinki University of Technology
P.O.Box 9800, FIN-02015 HUT, Finland
E-mail: {Ville.Tuulos, Petri.Myllymaki}@cs.helsinki.fi

***Abstract*— This work is part of a proactive information retrieval project that aims at estimating relevance from implicit user feedback. The noisy feedback signal needs to be complemented with all available information, and textual content is one of the natural sources. Here we take the first steps by investigating whether this source is at all useful in the challenging setting of estimating the relevance of a new document based on only few samples with known relevance. It turns out that even sophisticated unsupervised methods like multinomial PCA (or Latent Dirichlet Allocation) cannot help much. By contrast, feature extraction supervised by relevant auxiliary data may help.**

I. Introduction

In *proactive information retrieval*, the system adapts to the interests of the user that are inferred from implicit feedback. Feedback by explicitly indicating which documents are relevant to the user is naturally more accurate but the users often consider it too laborious and time-consuming. The usability and accuracy of information retrieval applications would be greatly enhanced by complementing explicit feedback with implicit feedback signals measured from the user and the interface. Research on implicit feedback potentially has even wider-ranging implications. If the feedback signal is reliable enough, it will be useful in a range of other applications as well. Ultimately, a genuine personal assistant could adapt to the goals and interests of the user and learn to disambiguate her vague commands and anticipate her actions.

In this first stage of the work we start with a simplified setting, where the user is reading a given document. That is, we assume that the document has already been chosen in some way or is a new one, and the task is to estimate whether it is relevant or not. This will be done using implicit feedback such as eye movements, which we studied in [1].

The problem with implicit relevance signals is that they will necessarily be noisy, and need to be complemented with any available sources of relevant information. Textual content is of course a natural one since it is the basis of all standard information retrieval. In this paper we study how accurately relevance can be estimated based on textual content only, when only few documents with known relevance are available. If textual content helps in prediction (compared to random performance), it will be used as prior knowledge in inferring relevance from implicit feedback. This problem is closely related to standard text classification, and some simple standard methods will be included in the comparisons.

Here we report the results of a feasibility study that aims at answering the following research questions: (1) is the prediction accuracy high enough, (2) whether a rigorous unsupervised model of the document collection will help in the task, and (3) whether suitable auxiliary data will help.

II. Setting

In this paper we focus on the following setting. Let D denote a collection of I documents $D_1, \ldots, D_I$. Each document D_i consists of words w, and the number of different possible words is J. In the following we make the standard simplifying "bag-of-words" assumption. The order of the words within a single document is considered irrelevant, and only the counts of different words in a document are used as features.

A lot of research related to this type of a setting is focused on unsupervised data exploration tasks like data clustering or dimensionality reduction. In data clustering the document collection D is partitioned into several subsets such that the documents within each subset are in some sense similar to each other, and different from the documents in the other subsets. In dimensionality reduction the goal is to find a low-dimensional representation of the document collection so that the coordinates of the resulting low-dimensional space correspond to some interesting factors that cannot be directly observed in the data. The Websom system [2] is an example of both data clustering and dimensionality reduction.

The models produced by data clustering or dimensionality reduction methods can be used for unsupervised data exploration tasks where the goal is to achieve a better understanding of the regularities governing the domain where the data is from. However, it is obvious that this type of models can also be used for *information retrieval* tasks where the goal can, for example, be to find from a document collection D the document D_i that is the most similar to a given, previously unseen document or query D_{I+1}.

In this work we deviate from the standard unsupervised data exploration setting and consider the following *supervised* modeling problem. We assume that each text document is provided with some labels. For simplicity, let us assume that the labels are simply binary, and let us denote the label of document D_i by r_i. If $r_i = 1$, we say that the document is *relevant*, otherwise it is considered irrelevant.

The meaning of relevance depends of course of the semantic interpretation of the binary labels $r_1, \ldots, r_I$, and is subjective to the person doing the labeling. Consequently, label $r_i = 1$ could, for example, represent the fact that the person doing the labeling liked the text in document D_i, or that she liked the matter that the text is referring to. In any case, the task we are facing is now the following: given a document collection $D = \{D_1, \ldots, D_I, D_{I+1}\}$, and the corresponding relevance labels $\{r_1, \ldots, r_I\}$ for all the documents except the last, infer the relevance of the last document D_{I+1}. Note that this setting differs from standard information retrieval, in that we are not searching relevant documents from D but instead want to predict relevance of a given new document D_{I+1}.

It should be noted that the task given above is supervised in the sense that all we are interested in is predicting the value of r_{I+1}, the relevance of the unlabeled document — we are not necessarily interested in understanding the deeper structure of the domain if that does not help us in our supervised prediction task. Of course, one can first build an unsupervised model of the problem domain and then use that model in the prediction task, and as a matter of fact, that is one of the approaches explored in this paper. However, as discussed and demonstrated in [3], [4], one should acknowledge that in this approach we are faced with the danger that the domain model only represents those regularities that are irrelevant with respect to the supervised prediction task, in which case the prediction task becomes impossible to solve.

As already noted, the relevances $\{r_1, \ldots, r_I\}$ are subjective. In a more general setting one could assume to have a relevance vector for each document, consisting of the relevance labels given by several individuals. In this case one could then use *collaborative filtering* [5] techniques in our supervised prediction task. However, in this paper we restrict ourselves to the *single user* case, where this type of techniques cannot be exploited.

If we restrict ourselves to simple binary labels as above, the prediction problem we are addressing is similar to the problem of *e-mail spam filtering*, where the goal is to distinguish useful e-mail messages from uninteresting ads, viruses and such. However, in this case the relevance of a document can be considered objective, not subjective, as most people seem to agree upon what is spam and what is not. This means that the amount of available data in spam filtering tasks is typically huge, whereas we in our single-user setting need to work with relatively small data sets. On the other hand, the spam filtering task can be considered relatively easy as the contents of the spam messages typically contain certain elements — for example, key words like “offer”, “viagra”, etc. — so that detecting these messages is easy, while we address problem domains where the textual contents give only very weak signals of the relevance of the document. A typical example of such a domain is the movie database discussed in the next section.

III. Data

We experimented with a data set of labeled text documents. The user-specific labels were collected from a movie rating database, where people have given ratings to movies according to their likes and dislikes. The textual descriptions of the movies were retrieved from the Internet, and the users’ ratings were associated to these text documents. Given a set of subjectively labeled documents of an individual user we build a model for this particular user’s relevances and use the model to predict the relevance of a new document.

This specific data set was chosen because of its size; it contained more than 70,000 users and 2 million ratings. We will later combine the ratings of different users by modeling user groups.

A. Original Data

The data was collected from a publicly available database of people’s ratings for a set of movies (EachMovie) [6]. Synopses of a set of 1398 movies were gathered from the Allmovie database [7] and they were used as the text documents. The ratings in EachMovie database had been gathered in a scale from one to five stars.

B. Preprocessed Data

Low level preprocessing included removing words inside brackets “()”, which were typically names, and stemming according to Porter’s algorithm [8]. Terms were required to appear at least 5 times in the document collection, which resulted altogether in 4619 terms.

We gathered a data set that conforms to our assumption of binary labels of “relevance” by picking up, for each user, the 10% of the movies with the best ratings (“relevant”), and the 10% with the lowest ratings (“irrelevant”). This has the additional desirable consequence that the originally possibly very different rating scales of different users become normalized. In this data set, the success probability of random guessing will be 50%.

Finally, we only accepted those users who had at least 80 ratings after this filtering. The resulting number of users was 134.

C. Term Selection

To reduce the dimensionality of the term space, 1000 terms were selected with the Odds Ratio algorithm [9] as described in Section IV-A.4. In some of the experiments, the set was reduced further to 500 terms (LDA500) by filtering with Linear Discriminant Analysis as described in the same section.

D. Auxiliary Data About Movie Genres

There was also a classification of the movies into 10 genres available in the EachMovie database. This classification was utilized in some of the experiments (details below). The genres were: Action, Animation, Art_Foreign, Classic, Comedy, Drama, Family, Horror, Romance and Thriller. Each movie typically belongs to 1 or 2 genres.

IV. Methods

The methods we used for estimating relevance consist of two stages. First, a representation for the document was formed, and then the relevance was predicted based on this representation. A few alternatives were tried for each stage; they vary in the degree of sophistication and in what kind of data they use for optimizing the predictions. The methods were tested on leave-out data as described in Section V.

A. Representation of Documents

For computational simplicity, all methods are based on the bag-of-words assumption: the order of the words is neglected.

1) Simple Unsupervised Features: The simplest representation is a binary term vector, where the entry corresponding to term w_j is zero if the term does not occur in the document, and one if it does.

The next, slightly more complex alternative would be to replace the binary numbers by frequency counts, or some simple functions of them, as in the standard "vector-space model" of information retrieval. In preliminary experiments this did not improve the results—probably because the most important terms rarely occur multiple times in our short documents—and we decided to use the binary vectors as the simplest alternative.

2) Unsupervised Feature Extraction with Multinomial PCA: An alternative method that takes the frequency of occurrence of words into account in a rigorous probabilistic fashion, starts from a J-component vector $\mathbf{w}_i$, where the jth component of $\mathbf{w}_i$ gives the number of occurrences of word w_j in document D_i. In the *multinomial PCA* (mPCA) approach [10] the document collection is modeled by assuming that the words are generated from K probability distributions, where K is a much smaller number than the number of words J. Each of these K probability distributions can be represented as a J-component vector where the jth component gives the probability for the occurrence of word w_j. As these probability distributions define which words occur together with high probability, they are often called "topics".

Let Ω denote a $J \times K$ matrix, where the jth column gives the probabilities for term w_j in each of the K topic distributions. Now, intuitively it makes sense that a textual document may contain text from several topic distributions, that is, a single document can be related to several different topics. In the mPCA approach this is modeled by assuming that the text generating probability distribution for each document is a weighted linear combination of all the topic distributions. More formally,

$$\mathbf{w}_i \sim \text{Multinomial}(\boldsymbol{\theta}_i \Omega, L_i), \tag{1}$$

where L_i denotes the number of words in document D_i, and $\boldsymbol{\theta}_i$ gives the mixing coefficients of the text generating probability distribution corresponding to document D_i. The prior distribution for the vectors $\boldsymbol{\theta}_i$ is usually assumed to be the Dirichlet distribution, the conjugate distribution of the multinomial.

Intuitively, the components of the vector $\boldsymbol{\theta}_i$ reveal to what extent document D_i addresses each of the topics. Consequently, as discussed in [11], mPCA can be seen as a multi-faceted clustering method, where each document belongs to each cluster (topic) with some probability. On the other hand, the model can also be viewed as a dimensionality reduction scheme: for those familiar with standard principal component analysis (see [12]), it is evident that the above model is a discrete equivalent for the standard PCA with the Gaussian data generating function replaced by the multinomial. It should be noted that although technically possible, it does not make rigorously sense to apply the PCA model directly to textual data, as the discrete text data is typically very non-normally distributed.

In summary, so far we have three different representations of text documents $D_1, \ldots, D_I$. First, they can be seen as strings of words. Second, ignoring the ordering of the words, they can be thought of as word count vectors $\mathbf{w}_1, \ldots, \mathbf{w}_I$ (and in the experiments we will further simplify them to binary vectors). Third, they can be treated as topic probability vectors $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_I$. We used these topic probability vectors $\boldsymbol{\theta}_i$ as feature vectors for the classification. To see how the mPCA model can be used for tasks like information retrieval, see for example [13].

3) Given Supervised Features: For comparison, we also used the movie genres assigned to each movie (see Section III-D). Documents were coded as binary vectors of these features, where each component of the vector corresponds to a genre. The components of these 10-dimensional vectors indicate to which genres the document belongs to.

The genre assignments have been carefully chosen to describe the movies and hence they are expected to be better features than the very noisy texts. Since the genres are not known for new documents, however, they do not solve our problem but they will be used as a kind of measure for "best possible performance."

4) Genre-Based Feature Extraction and Linear Discriminant Analysis: Odds Ratio algorithm [9] was used to initially reduce the number of terms to 1000 that discriminate between the given movie genres. The Odds Ratio is defined as

$$OR(w_k, c) = \frac{P(w_k|c)\ (1 - P(w_k|\neg c))}{(1 - P(w_k|c))\ P(w_k|\neg c)} \tag{2}$$

where w_k is a term, $P(w_k|c)$ is the frequency-based estimate of the probability that term w_k occurs in a document of class c and $\neg c$ is the complement of class c. Terms that had the highest Odds Ratio on the average were selected.

In some of the experiments we further reduced the dimensionality with Linear Discriminant Analysis (LDA), a classical linear classification method (see [14]). It finds a projection that best discriminates between the classes, and for two-class case the projection is onto a one-dimensional feature. Since our classes are non-exclusive, that is, each movie may belong to several genres, we sought one feature for each genre, to discriminate between movies belonging and not belonging to it. As a result we got 10 discriminative features. Projection of the term vectors on these directions yielded a 10-dimensional feature space (LDAproj).

The LDA assumes that the given classes are normally distributed with equal covariance matrices. This clearly does not hold for our data, but it turned out that the discrimination still succeeded well.

In other experiments we also used LDA to reduce the dimensionality of the binary term space; from the 1000 terms we chose those 500 terms (LDA500) that had the greatest overall loadings on the discriminative directions.

B. Classification

Two simple but powerful methods, the log-linear model and the K-Nearest-Neighbor classifier, were used for the final classification to relevant and irrelevant documents using different vectorial representations of documents. The classification was done for each user separately. A spam-filtering algorithm was used as a baseline method for the classification.

1) Log-linear Model: The log-linear classifier was used to model the relevances of each user. The input $\mathbf{x}_i$ denotes one of the vectorial representations for the document D_i, for instance a binary term vector. The probability of document D_i to be relevant ($r_i = 1$) to the user is assumed Bernoulli distributed with document-specific mean $\mu_i(\mathbf{x}_i)$,

$$p(r_i \mid \mathbf{x}_i) = \mu_i^{r_i}(1-\mu_i)^{1-r_i}. \tag{3}$$

The logit of the mean is assumed to obey a linear model with user-specific parameters $\mathbf{c}$:

$$\text{logit}(\mu) = \text{logit}(E[r \mid \mathbf{x}]) = \mathbf{c}^T\mathbf{x}. \tag{4}$$

The parameters $\mathbf{c}$ are sought by maximizing the likelihood of the observed data, i.e., ratings of the individual user. For details of optimization see [15]. Predicted relevance of a new document $\mathbf{x}_{new}$ in this model is $\text{logit}^{-1}(\mathbf{c}^T\mathbf{x}_{new}) \in [0,1]$. In the tests the predictions were rounded to binary predictions $\in \{0,1\}$.

2) K-Nearest-Neighbor Classification: K-nearest-neighbor classifier (KNN) stores a reference set of labeled samples. A new unlabeled sample is classified according to a majority vote of its K nearest neighbors in the reference set. The size of the neighborhood is a free parameter, and the distance measure that defines the neighborhood needs to be chosen as well. We used Euclidean distances since our preliminary tests did not show marked differences in the results for the other metric considered (Hellinger distance [16]).

3) Spam Filtering Method: A state-of-the-art spam filtering algorithm, CRM114 [17], was used as a baseline method. CRM114 works by sliding a five-word window over the document. Each window increases the frequency counts of the corresponding words. Finally, a Naive-Bayes classifier based on empirical the frequencies gives the classification.

C. Experimental Setting

The classification was initially computed with both the log-linear model and the K-nearest neighbor classifier with $K = 9$, but since the log-linear model consistently performed better, the KNN results were left out of the discussion. All the models were trained for each user separately and tested with leave-one-out crossvalidation. Mean prediction error over each user's predictions was taken as a user-specific error, and mean prediction error over all users was used as performance measure between models. Since all the users had equal numbers of relevant ($r = 1$) and irrelevant ($r = 0$) ratings, prediction error of 0.50 corresponds to random guessing.

V. Results

A. Comparison of Unsupervised Feature Extraction Methods

Our first hypothesis was that a multinomial PCA (mPCA), computed from the whole text collection, would find useful topics that would help reducing noise in the texts and help in predicting relevance. We compared mPCA-based feature extraction with the completely unsupervised spam filter, and with binary term vectors. To get an estimate of a lower limit for the prediction error, we further included genre vectors that are supposed to be superior to the other features.

In detail, the experiments were carried out as follows. **mPCA**: The number of topics was fixed to 10, and the output of the mPCA model was a point estimate of the topic distribution $\boldsymbol{\theta}$ for each document. The log-linear model was fitted for each user in this topic space. **genre**: The log-linear model was fitted to the genre vectors of each user. **LDA500**: The binary term vectors are not strictly speaking unsupervised, since the set of terms was reduced, for computational reasons, with a partly supervised method (LDA500 described in section IV-A.4). A log-linear model was fit to the term vectors. **crm114**: a state-of-the-art spam filtering algorithm [17].

The results shown in Figure 1 reveal that the term vector-based classification (LDA500) does not differ from that obtained by chance. The spam filter (crm114) is slightly and mPCA considerably better, but both are far from the performance of the supervised genre vector.

The reason for the weak performance of the spam filter is probably that it has been designed for a different task. Typical spam is relatively homogenous and there is plenty of training material available. Hence, there is no need to optimize the performance of the filter for very small data sets, such as our small user-specific sets.

The mPCA feature extraction was clearly better than the binary term vectors but still far from the "best possible performance" of the genre vectors. Note, though, that at this stage of the experiments it was of course not clear whether the performance of genre vectors could be reached by texts only, and we were simply searching for the limits.

The mPCA was included to reduce the dimensionality. It was, however, optimized in a purely unsupervised fashion, and there is no theoretical reason why it should help in our discriminative task. It should help if the variation it models is useful for discrimination but otherwise not. So the main question was whether the bad performance was due to overfitting of the log-linear model or that the mPCA loses the information required for the classification. We checked this by computing the performance on the training set (Table I).

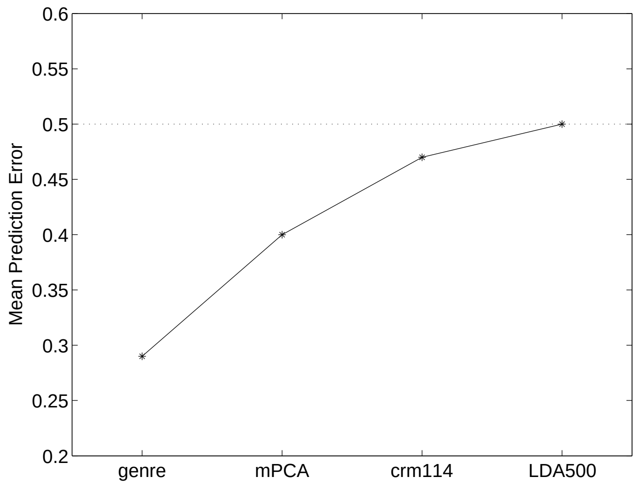

Fig. 1. Classification errors for predicting relevance of left-out documents with a log-linear model, based on 4 different feature sets. **genre:** Binary genre vector. **mPCA:** posterior estimates of mPCA-topic probabilities. **crm114:** Spam filter CRM114. **LDA500:** Binary term vector. Dotted line: random performance.

Since the performance on the training set was clearly better than on the test set, the tentative conclusion was that the mPCA does not lose all relevant information but that there still was too much variation in the result even after the mPCA-based dimensionality reduction. The few labeled samples are not sufficient for building reliable predictors in the mPCA space.

TABLE I

DIFFERENCE OF THE PERFORMANCE OF THE MPCA FEATURE EXTRACTION IN THE TRAINING AND TEST SETS. THE FIGURES ARE MEAN PREDICTION ERRORS IN LEAVE-ONE-OUT CROSSVALIDATION.

	Train set	**Test set**
mPCA	0.31	0.40

B. *Feature Extraction Supervised with Auxiliary Genre Data*

The conclusion from the previous section is that the number of labeled samples was too small. On the other hand, we know that prediction based on the genre vectors was more successful, and there are plenty of texts available with known genres. Hence, the next idea was to supervise the feature extraction by the genre vectors: Optimize such a feature extractor for texts that it would give good predictions of the genres. This will reduce the dimensionality of the term space, and a classifier in this reduced-dimensional space might perform better in predicting relevance. Such a feature extractor would be applicable to new documents with unknown genres as well.

Linear Discriminant Analysis was used to obtain one discriminative direction for each genre in the term space, and a new document was then projected onto this 10-dimensional feature space as described in Section IV-A.4. The results (LDAproj) of predicting relevance with a log-linear model in this space are shown in Figure 2. The genre-supervised features give almost the same performance as the original genre vectors.

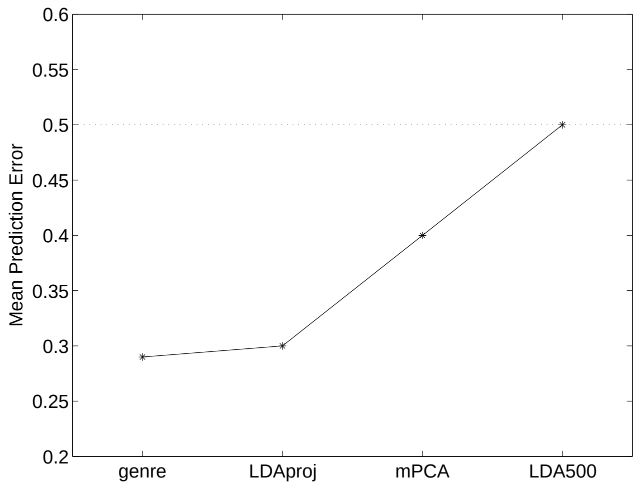

Fig. 2. Genre-supervised LDA-features (**LDAproj**) perform well. For description of the other features see Figure 1.

Finally, we checked whether selecting the terms discriminatively before training mPCA would lead to any improvement, but it led to overfitting as well.

VI. DISCUSSION

In this first feasibility study we have investigated prediction of relevance of a given document based on only a small set with known relevance. It turned out that a completely unsupervised multinomial PCA model of the whole document collection helped somewhat. If suitable auxiliary data is available for a larger set of documents, here the genre classifications, it can be used to help reduce the problem of small data sets. Supervising feature selection by the genres improved performance of subsequent prediction of relevance.

In this work we focused on single-user systems and did not combine the models optimized for different users. Such collaborative filtering will be studied later, and the models will additionally be combined with models of implicit feedback for proactive information retrieval.

ACKNOWLEDGMENT

This work was supported by the Academy of Finland, decision no. 79017, and by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views. The authors would like to thank Drs Kai Puolamäki and Janne Sinkkonen, and all other people in the PRIMA project for useful discussions, and acknowledge that access rights to the data sets and other materials produced in the PRIMA project are restricted due to other commitments.

REFERENCES

[1] J. Salojärvi, I. Kojo, J. Simola, and S. Kaski, "Can relevance be inferred from eye movements in information retrieval?" in *Proceedings of the Workshop on Self-Organizing Maps (WSOM'03)*, Hibikino, Kitakyushu, Japan, September 2003, pp. 261–266.

[2] T. Kohonen, S. Kaski, K. Lagus, J. Salojärvi, J. Honkela, V. Paatero, and A. Saarela, "Self organization of a massive document collection," *IEEE Transactions on Neural Networks*, vol. 11, pp. 574–585, 2000.

[3] P. Kontkanen, P. Myllymäki, T. Silander, and H. Tirri, "On supervised selection of Bayesian networks," in *Proceedings of the 15th International Conference on Uncertainty in Artificial Intelligence (UAI'99)*, K. Laskey and H. Prade, Eds., 1999, pp. 334–342.

[4] R. Cowell, "On searching for optimal classifiers among Bayesian networks," in *Proceedings of the Eighth International Conference on Artificial Intelligence and Statistics*, T. Jaakkola and T. Richardson, Eds., 2001, pp. 175–180.

[5] J. Griffith and C. O'Riordan, "Collaborative filtering," National University of Ireland, Galway, Tech. Rep. NUIG-IT-160900, 2000.

[6] EachMovie movie rating database. [Online]. Available: http://research.compaq.com/SRC/eachmovie/

[7] Allmovie database containing movie information. [Online]. Available: http://www.allmovie.com

[8] M. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[9] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, vol. 34, no. 1, pp. 1–47, 2002.

[10] W. Buntine, "Variational extensions to EM and multinomial PCA," in *Proceedings of the 13th European Conference on Machine Learning*, ser. Lecture Notes in Artificial Intelligence, T. Elomaa, H. Mannila, and H. Toivonen, Eds., vol. 2430. Springer-Verlag, 2002, pp. 23–34.

[11] W. .Buntine and S. Perttu, "Is multinomial PCA multi-faceted clustering or dimensionality reduction?" in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, C. Bishop and B. Frey, Eds. Society for Artificial Intelligence and Statistics, 2003, pp. 300–307.

[12] M. Tipping and C. Bishop, "Mixtures of probabilistic principal component analysers," *Neural Computation*, vol. 11, pp. 443–482, 1999.

[13] W. Buntine, P. Myllymäki, and S. Perttu, "Language models for intelligent search using multinomial PCA," in *Proceedings of the First European Web Mining Forum at the 14th European Conference on Machine Learning and the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, B. Berendt, D. M. A. Hotho, M. van Someren, M. Spiliopoulou, and G. Stumme, Eds. Ruder Boskovic Institute, 2003, pp. 37–50.

[14] N. H. Timm, *Applied Multivariate Analysis*. New York: Springer, 2002.

[15] I. Nabney, "Efficient training of RBF networks for classication," in *Proc. ICANN'99*, 1999.

[16] M. Fannes and P. Spincemaille, "The mutual affinity of random measures," 2001. [Online]. Available: http://arXiv.org/abs/math-ph/?0112034v1

[17] W. Yerazunis, "Sparse binary polynomial hashing and the CRM114 discriminator," In the Web. [Online]. Available: http://crm114.sourceforge.net/CRM114_paper.html