

Document organization and retrieval using self organizing maps and statistical language modeling

A. Georgakis, C. Kotropoulos, A. Xafopoulos, and I. Pitas *

Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki 54006, Greece
{costas, pitas}@zeus.csd.auth.gr

Abstract. In this paper we present a method for document organization and retrieval based on statistical language modeling. The proposed method, which is based on the vector model, uses nonlinear interpolation to provide more accurate statistical estimators of the conditional probabilities employed for encoding the context of each word. An information retrieval system is built using the self-organizing map algorithm. In the first step, the self-organizing architecture is used to cluster the feature vectors and to build clusters of semantically related words. Subsequently, the collection of documents is encoded into vectors and the same algorithm is used to cluster the documents in contextually related classes. The information retrieval system is queried using a sample document and the corresponding precision-recall curve is provided.

1 Introduction

Information retrieval (IR) is a difficult task for which a perfect solution has not been found yet. IR consists of two distinct steps: *organization* and *retrieval*. Organization refers to the representation and storage of the available data, whereas retrieval refers to the exploration of the organized data through the use of specific queries [1]. Prior to retrieval, the data repository, which in the case under consideration is a collection of documents called *corpus*, has to be organized according to the retrieval method to be applied. Without a retrieval-oriented organization method the retrieval of even a fraction of the available documents becomes onerous [1].

The traditional IR systems rely on index terms in order to index the documents and use the set theory and the Boolean algebra for retrieval purposes. A query aiming at exploring these documents must fulfil certain criteria; it is formulated using a limited number of keywords which are combined together using Boolean operators. The above described organization and retrieval system is usually referred to as the *Boolean* model. The basic drawbacks of the Boolean model are the following:

* This work was supported by the European Union IST Project "HYPERGEO: Easy and friendly access to geographic information for mobile users" (IST-1999-11641).

- It does not support weighting of the terms of the query based on their significance.
- Users are not familiar with the Boolean logic.
- No sufficient mechanism for ranking of the retrieved documents exists.

A solution to the ranking problem derives from the relaxation of the conditions posed by the Boolean operators with the usage of a set of *fuzzy* operators [1]. Unfortunately, the weighting problem persists.

An alternative to the Boolean model stems from the following fact; given a user generated query, the retrieval system should be able to rank the references in the collection of documents according to their probability of relevance to the request. This leads to the conclusion that there is a set of relevant and a set of non-relevant documents in the collection. Through iteratively interaction with the user, the IR system tries to infer the probability that the user will find a retrieved document relevant. This model is the so-called *probabilistic* model. The major drawbacks of the probabilistic model are the initial guessing of the ideal set and the inability of the model to take into account the frequency of the index terms inside the query.

A further constraint regarding these retrieval systems is the fact that they are capable of handling only keyword-based queries which introduces serious restrictions; the formulation of such queries is not always feasible. For example, let consider a user willing to retrieve documents on the basis of a user-defined document. Due to the limitations of the retrieval system the user would have to formulate a query from specific terms extracted from the template document. A more natural approach would be the use of the whole document as input to the retrieval system.

The aforementioned shortcomings are easily resolved by the *vector* model. In this model, the documents and the queries are encoded into vectors. Owing to the available vector norms, the documents are easily clustered into contextually relative collections. The similarities between the query and the documents are measured using vector norms and the retrieved documents are sorted according to the same norm. Furthermore, the proposed model is user-friendly and supports term-weighting.

The outline of the paper is as follows. In Section 2 we propose a variant of the standard statistical language model based on nonlinear interpolation techniques so as to provide more reliable estimates of word sequences. Section 3 provides a brief description of the SOM architecture. Section 4 deals with the problem of the dimensionality of input vectors computed with the proposed variant while in Section 5.3 the mean squared error and recall-precision curves for both models are presented.

2 Language Modeling

Language modeling can be seen as a case of statistical inference, which makes inferences about the unknown distribution of data [2]. An approach divides this problem into three possibly overlapping areas. Division of training data into

equivalence classes, estimator selection for the classes and combination of estimators. A remarkable number of language modeling techniques have emerged, e.g. skipping, clustering, caching, higher-order n -grams, tree-based models, lattice-based models and smoothing [3]. A combination of language models can be found in [3]. Both speech recognition [4, 5] and information retrieval (IR) [6] resort to language modeling.

Let $V = \{w_1, w_2, \dots, w_k\}$ denote the vocabulary of distinct word symbols having size $|V| = k$. The *a priori* probability $P(w_1^m) = P(w_1, w_2, \dots, w_m)$ for the word sequence $w_1^m = w_1 w_2 \dots w_m$ with $w_i \in V$ and $w_i^i = w_i$ can be expressed as a product of the conditional probabilities $P(w_i | w_1 w_2 \dots w_{i-1}) = P(w_i | w_1^{i-1})$, using the chain rule:

$$P(w_1^m) = P(w_1) \cdot \prod_{i=2}^m P(w_i | w_1^{i-1}). \quad (1)$$

Accordingly, the task of language modeling is reduced to the estimation of the probabilities appearing on the right-hand side of (1). In this paper, the above word sequence w_1^{i-1} is referred to as the $(i-1)$ -length backward history h_{i-1} of the underlying stochastic process for $P(w_1^m)$, where $h_n \in V^n$, $n \in \mathbb{N}^+$. On the other hand, the word w_i denotes the prediction [7]. All possible conditional probabilities that the language model estimates are the parameters of the model, which are $|V|^n$ in number¹.

Related to the division of training data into classes is the type of models used. A widely used language model is the n -gram model. When using this model for $n \in \mathbb{N}^+$, the probability $P(w_1^m)$ can be approximated by restricting the history to the preceding $n-1$ words, except for the first few words of the sequence for which fewer than $n-1$ words exist:

$$P_{(n)}(w_1^m) = P(w_1) \cdot \prod_{i=2}^m P(w_i | w_{\max\{i-n+1, 1\}}^{i-1}) = P(w_1) \cdot \prod_{i=2}^m P(w_i | h_{\min\{n-1, i-1\}}). \quad (2)$$

Generally $w_i^j = \emptyset$ when $j < i$ and $h_l = \emptyset$ when $l < 1$. Also $P(w|\emptyset) = P(w)$. Another moot point is the value of n , which is related to the number of equivalence word bins. Higher n values result in more bins and the opposite. The whole thing constitutes a reliability-discrimination compromise. Generally higher n values require larger corpora in order to provide robust estimates, owing to the much larger number of model parameters that need to be estimated. In our experiments bigram and unigram models are used.

The next issue is the selection of an appropriate statistical estimator for the parameters of the n -gram models. The straightforward approach for the estimation of the conditional probabilities $P(w_i | w_{i-n+1}^{i-1})$ is the well-known maximum likelihood (ML) estimator, which uses the notion of relative frequencies [8]. If TC denotes the training data collection called the *training corpus*, the ML estimate

¹ To be more accurate the independent parameters are $|V|^n - 1$ since the sum of the probabilities equals 1.

of the latter conditional probabilities derived from TC is given by the number of occurrences of the word sequence constituted by the word w_i and its preceding (restricted) history $w_{i-n+1}^{i-1} = h_{n-1}$ in TC divided by the number of occurrences of this history:

$$\hat{P}_{ML(n)}(w_i | w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^i)}{c(w_{i-n+1}^{i-1})}, \quad 1 < n < i \quad (3)$$

where $c(w_i^j)$ is the number of occurrences of the word sequence w_i^j in TC. For bigram models (3) is simplified to:

$$\hat{P}_{ML(2)}(w | h_1) = \frac{c(h_1, w)}{c(h_1)}. \quad (4)$$

If N denotes the TC size, that is, the number of word tokens in TC, the unigram ML estimate is $\hat{P}_{ML(1)}(w_i) = \frac{c(w_i)}{N}$. For the zero-gram case, $\hat{P}_U(w_i) = \frac{1}{|V|}$ with U implying a uniform estimate.

Considering the number $|V|^n$ of potential n -grams in TC, especially when n increases, and the limited input in terms of both the lexical units and the syntactic limitations of TC, it is concluded that the training corpus only approximates a small percentage of the potential n -grams. This fact becomes even worse when natural languages, which have very large $|V|$, are being modeled. To deal with this sparsity problem of missing, over- and under-estimated n -grams, two methods are usually used, namely building equivalence word classes, and smoothing estimates. We shall confine ourselves to the frequently used smoothing methods.

Smoothing is a very useful technique used in the construction of robust language models. One smoothing method is the interpolation which performs a suitably weighted combination of n and lower order estimates [9]. During or after smoothing, the resulting probability estimates are normalized so that they sum to unity per vocabulary word. Several types of interpolation have been proposed in speech literature [2, 5, 9]. The nonlinear interpolation method lies among the best models. Let \bar{h}_n be the generalized history of length n which refers to histories of length less than n . We denote by $c_t(h_n)$ the number of n -grams which have exactly t occurrences and their history is h_n in TC. Accordingly, the unseen n -grams beginning with h_{n-1} are $c_0(h_{n-1})$ in total. The nonlinear interpolation (NLI) estimate of the backward conditional n -gram probabilities, for $n > 1$, is:

$$\hat{P}_{NLI(n)}(w | h_{n-1}) = \begin{cases} \frac{\max\{c(h_{n-1}, w) - \delta(h_{n-1}, w), 0\}}{c(h_{n-1})} + \\ + \frac{\delta(h_{n-1}, w)(|V| - c_0(h_{n-1}))}{c(h_{n-1})} \hat{P}_Q(w | \bar{h}_{n-1}), & \text{if } c(h_{n-1}, w) = 0 \wedge c(h_{n-1}) > 0 \\ \hat{P}_Q(w | \bar{h}_{n-1}), & \text{if } c(h_{n-1}) = 0 \end{cases} \quad (5)$$

where $\delta(h_{n-1}, w)$ denotes a discount parameter. The estimate $\hat{P}_Q(w | \bar{h}_{n-1})$ is a suitably selected one that takes into consideration a truncated generalized history. The parameter Q is usually replaced by the same estimate (in our case

NLI), ML or U and \bar{h}_{n-1} by h_{n-1} or \emptyset . Also, $\hat{P}(w|\emptyset) = \hat{P}(w)$. We use the above NLI estimate on bigrams and the parameters δ, Q are substituted as in:

$$\hat{P}_{NLI(2)}(w|h_1) = \frac{\max\{c(h_1, w) - D_{(2)}(h_1), 0\} + D_{(2)}(h_1)(|V| - c_0(h_1))\hat{P}_{ML(1)}(w)}{c(h_1)} \quad (6)$$

where $c(h_1)$ is non-zero due to the fact that all encountered words are included in V . For the discount parameter an ‘‘absolute model’’ is used where

$$D_{(2)}(h_1) = \frac{|V|b_{(2)}}{c_0(h_1)}, \quad b_{(2)} = \frac{c_{1(2)}}{c_{1(2)} + 2c_{2(2)}}$$

and $c_{t(n)}$ is the number of n -grams with exactly t occurrences in TC [9].

The conventional n -gram modeling is based on the backward context to model words. Throughout a text a forward word context can also be considered. We refer to the latter context as ‘‘forward’’ history, g . This history refers to a word sequence used as a condition and which succeeds a word w that is being predicted. The ‘‘forward’’ conditional bigram probability estimates for the ML and NLI cases can be calculated using the Bayes rule:

$$\hat{P}_{ML(2)}(w|g_1) = \frac{\hat{P}_{ML(2)}(g_1|w)\hat{P}_{ML(1)}(w)}{\hat{P}_{ML(1)}(g_1)}, \quad (7)$$

and

$$\hat{P}_{NLI(2)}(w|g_1) = \frac{\hat{P}_{NLI(2)}(g_1|w)\hat{P}_{ML(1)}(w)}{\hat{P}_{ML(1)}(g_1)}. \quad (8)$$

where for the unigrams ML estimates are used.

The above models are used to construct feature vectors used by the SOM algorithm and are based on the contextual information provided in the form of n -gram estimates. After the corpus processing (Section 5.1) each (word) stem in TC is modeled as a $3|V| \times 1$ vector \mathbf{x}_i of probability estimates. This vector consists of three $|V|$ -sized parts. These parts for a particular stem consider this stem as forward history g_1 of all other stems and as backward history h_1 of all other stems, respectively. The ML forward and backward estimates (4) and (7) in the case of ML (non-smoothed) bidirectional bigram models (ML-BBMs), and the NLI forward and backward estimates (6) and (8) in the case of NLI smoothed BBMs (NLI-BBMs) are the two ‘‘outer’’ parts applied on stems instead of whole words. The NLI conditional probabilities are normalized so that they sum to unity per history stem. The ‘‘inner’’ part is common for both models and consists of a scaled unigram ML estimate of the stem \mathbf{x}_i [10]:

$$\mathbf{x}_{i(ML)} = \begin{bmatrix} \sum_{l=1}^{|V|} \hat{P}_{ML(2)}(x_l | g_1 = x_i) \mathbf{e}_l \\ \epsilon \hat{P}_{ML(1)}(x_i) \mathbf{e}_i \\ \sum_{m=1}^{|V|} \hat{P}_{ML(2)}(x_m | h_1 = x_i) \mathbf{e}_m \end{bmatrix}, \quad \mathbf{x}_{i(NLI)} = \begin{bmatrix} \sum_{l=1}^{|V|} \hat{P}_{NLI(2)}(x_l | g_1 = x_i) \mathbf{e}_l \\ \epsilon \hat{P}_{ML(1)}(x_i) \mathbf{e}_i \\ \sum_{m=1}^{|V|} \hat{P}_{NLI(2)}(x_m | h_1 = x_i) \mathbf{e}_m \end{bmatrix} \quad (9)$$

where \mathbf{e}_i denotes the $|V| \times 1$ unit vector that has one in the i -th component and zeros elsewhere and ϵ is a small real constant, e.g. 0.2. Its purpose is to enhance the influence of the context part over the symbol part of a stem in topological ordering.

NLI-BBMs consist our proposal against ML-BBMs, which are commonly employed in the SOM algorithm [11] for IR purposes.

3 Self-organizing maps

SOMs are feedforward artificial neural networks (NN) with a single computational layer of neurons arranged on a two or three-dimensional lattice [11, 12]. SOMs are capable of forming a nonlinear mapping from an arbitrary dimensional data manifold onto the low-dimensional discrete map. In doing this, the algorithm takes into consideration the relations of the presented features and computes an optimal representation that approximates these data in the sense of some error criterion, usually the mean square error (MSE).

Let \mathcal{X} denote the set of normalized vector-valued observations $\{\mathbf{x}_m = (x_{1m}, x_{2m}, \dots, x_{N_w m}) \in \mathbb{R}^{N_w}\}$ where N_w is the dimension of the input space². By \mathcal{W} we denote the set of normalized time varying reference vectors $\{\mathbf{w}_i(t) \in \mathbb{R}^{N_w}, i = 1, \dots, K\}$, where the notion t denotes discrete time ($t \in \mathbb{N}$).

The SOM algorithm works as follows: firstly, the weight vectors of the neurons are randomly initialized. Then, the algorithm randomly selects a feature vector \mathbf{x}_m from the set \mathcal{X} and exhaustively searches the weight vectors $\mathbf{w}_i(t), i = 1, 2, \dots, K$, for the most correlated neuron, the so-called *winning* neuron. The index of the winning neuron is given by: $c = \arg \min \|\mathbf{x}_m - \mathbf{w}_i(t)\|$ where $\|\cdot\|$ denotes the Euclidean distance between the feature vector and the weight vector.

Afterward, both the weight vector of the winner and the weight vectors in its neighborhood are modified toward \mathbf{x}_m . The following equation describes the updating process:

$$\mathbf{w}_j(t+1) = \begin{cases} \mathbf{w}_j(t) + a_{cj}(t) [\mathbf{x}_m - \mathbf{w}_j(t)], & \forall j \in \mathcal{N}_c \\ \mathbf{w}_j(t), & \forall j \notin \mathcal{N}_c \end{cases} \quad (10)$$

² Section 4 provides the necessary details regarding the dimensionality of the feature vectors deriving from (9) and the dimensionality used throughout the rest of the paper.

where \mathcal{N}_c denotes the neighborhood centered on the winner and $\alpha_{cj}(t)$ is the so-called *neighborhood function*.

When the above described procedure is repeated iteratively an adequate number of times for each \mathbf{x}_m it eventually leads to a global ordering of the feature space on the lattice.

4 Dimensionality reduction

The vectors derived from (9) are in the $3|V|$ -dimensional space, whose dimensionality is exceptionally high. This problem is tackled by using any dimensionality reduction technique. For example, a reduction to N_w ($N_w \ll 3|V|$) is achieved by the linear projection $\tilde{\mathbf{x}}_m = \Phi \mathbf{x}_m$ where $\tilde{\mathbf{x}}_m$ and \mathbf{x}_m are the projected and original vectors, respectively and Φ is the $N_w \times 3|V|$ matrix of the linear mapping. Kaski *et. al.* suggested a suboptimal approach to the previous problem using a random matrix Φ that has the following properties [13]:

- The components in each column are chosen to be independent, identically distributed Gaussian variables with zero mean and unit variance.
- Each column is normalized to unit norm.

In this paper N_w is assumed equal to 300.

Afterward, the sample variance of the m th component in the feature vector is computed using:

$$u_m = \sum_{j=1}^{|V|} (x_{mj} - \bar{\mathbf{x}}_m)^2, \quad m = 1, 2, \dots, N_w \quad (11)$$

where $\bar{\mathbf{x}}_m = \frac{1}{|V|} \sum_{j=1}^{|V|} x_{mj}$ denotes the sample mean and $|V|$ is the number of feature vectors.

The components of the feature vectors \mathbf{x}_m and the neuron weights are rearranged in descending order with respect to their sample variance u_m . Finally, the Euclidean distance used in identifying the winner is decomposed in the following way:

$$\|\tilde{\mathbf{x}}_m - \mathbf{w}_i(t)\| = \sum_{j=1}^{d'} (\tilde{x}_{(j)m} - w_{(j)i}(t))^2 + \sum_{j=d'+1}^{N_w} (\tilde{x}_{(j)m} - w_{(j)i}(t))^2 \quad (12)$$

where $\tilde{x}_{(j)m}$ and $w_{(j)i}(t)$ denotes j -th component of the rearranged (ordered) feature vector and weight vector correspondingly. Furthermore, d' is an arbitrary number, $d' < N_w$. The first sum in (12) contains the components of the feature vectors with the strongest values of the sample variance u_m , whereas the second sum contains the components whose impact in the selection of the winning neuron is more or less the same for every feature vector. By carefully selecting the parameter d' and omitting the second sum in (12) one can get an accurate estimation of the winning neuron. Figure 1 presents the percentage of

the successfully identified winner neuron in respect to the dimensionality difference which results from the selection of a positive value for the parameter d' . It can be seen that the performance of the suggested technique is satisfactory for a further reduction of the space dimensionality up to half of the initial dimension.

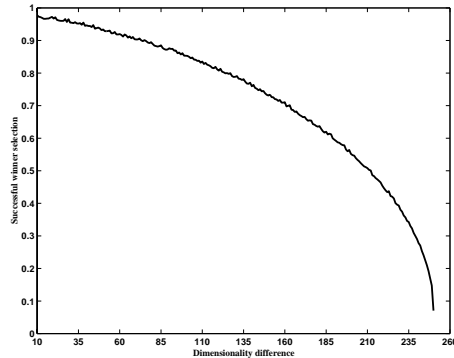


Fig. 1. Percentage of successfully identified winning neurons in the formation of the word category map with respect to the dimensionality difference $N_w - d'$.

5 Application

5.1 Corpus Processing

Throughout the training process a TC comprising 650 full-text HTML files was used. It contained nearly 230,000 words (word tokens), which were manually collected over the Internet. The HTML files are web pages of touristic content and in its current state is biased in the sense that web pages related to Greece, Spain and Germany form the majority [14]. The selected files are annotated by dividing them into 18 categories related to tourism such as accommodation, history, geography etc., so that ground truth is incorporated.

Before testing both language models a series of actions had to be taken in order to be feasible to use (9). The first step deals with *HTML* as well as *plain text* cleaning. HTML cleaning refers to the removal of the HTML tags and entities, and the appropriate treatment of some special tags, while plain text cleaning refers to the removal of URLs, email addresses, numbers, punctuation marks and the formation of word tokens. The sole punctuation mark left intact is the full stop, providing a rough sentence delimiter. Text cleaning also includes the removal of some common English words (such as articles, determiners, prepositions, pronouns, conjunctions, complementizers, abbreviations) and some non-English frequent terms in a processing step called *stopping*. The aforementioned step resulted in a corpus of $N = 125,000$ word tokens (training instances).

Subsequently, *stemming* is performed. Stemming refers to the elimination of word suffixes so that the resultant vocabulary shrinks, though keeping the informative context of the text. The underlying assumption for the successful usage of a stemming program, called a stemmer, is that morphological variants of words are semantically related [15]. The application of the commonly used Porter stemmer [16] resulted in a vocabulary size of $|V| \simeq 8700$ stem types (distinct occurrences).

5.2 Training

After the encoding of the words appearing in TC into vectors using (9), these vectors are clustered using the SOM algorithm in an effort to build clusters of semantically related words. This is based on empirical and theoretical observations that words semantically relative have more or less the same preceding and succeeding words. The result of this step is the so-called *word category map* (WCM)[12]. Grey levels near 255 imply that fewer word stems have been assigned to those particular neurons whereas those near 0 mean larger word densities. Figure 2 depicts the WCM and a neuron which is labeled by the words *apartment* and *room*. In the second panel of the *Results* window can be see the documents that contained these words.

The final step towards the vector model is the clustering of the documents. For each of the documents in TC, a normalized histogram of word categories is computed to derive the so-called *document vector* a_k . Subsequently, the NN is trained again using the document vectors resulting in the so-called *document map* (DM). The resulting map is expected to contain clusters of contextually related documents. Figure 3 depicts the resulting clusters in a section of the map which is labeled by documents relevant to regions of Spain.

5.3 Retrieval

In order to test the suggested language model a series of tests were conducted. The NN was trained twice. Primarily using ML-BBMs in calculating the feature vectors and secondly using NLI-BBMs. Our experiments revealed that the proposed variant leads to more homogeneous clusters than the standard model. This conclusion is drawn by the fact that the mean squared error (MSE) for NLI-BBMs was smaller than the MSE for ML-BBMs during the formation of the WCM throughout the training phase, as depicted in Fig. 4a. Furthermore, the values of the MSE for the proposed variant was nearly 28% lower than the standard model during the first iterations of the algorithm. Finally, for the standard model the number of training iterations needed so that the MSE drops to the $\frac{1}{e}$ of the initial value was about 40% higher than the proposed variant.

The quality of the clusters is measured by querying the retrieval system using a sample test document. The system retrieves the training corpus documents that are represented by the best matching neuron of the DM. The training documents retrieved are ranked according to their Euclidean distance from the sample test document. Subsequently they are classified as either being relevant

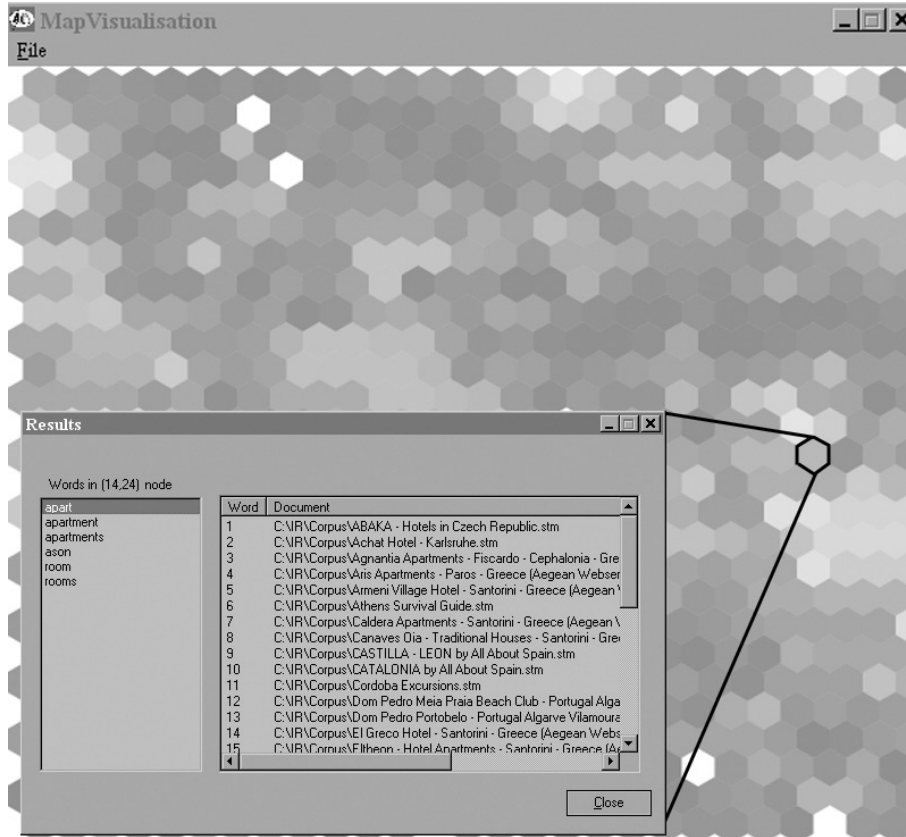


Fig. 2. The resulting word category map for a 27×27 neural network.

to the sample test document according to their annotation or not. Table 1 is the 2×2 contingency table which shows how the training corpus is divided.

For both techniques the *precision-recall* curve is calculated [17]. Precision is defined as the proportion of retrieved documents that are relevant: $P = r/n_2$ where r denotes the number of relevant documents which are retrieved and n_2 denotes the number of retrieved documents. Recall is the proportion of relevant documents that are retrieved: $R = r/n_1$ where n_1 is the total number of relevant documents in the corpus. Figure 4b depicts the recall - precision curves of the recall phase for both models. It is obvious that NLI-BBMs lead toward more uniform clusters.

6 Conclusion

The need for a robust language model for IR purposes motivated us to apply a smoothing method, namely, the nonlinear interpolation for the conditional probabilities. The resulted modeling (NLI-BBMs) was tested against the standard

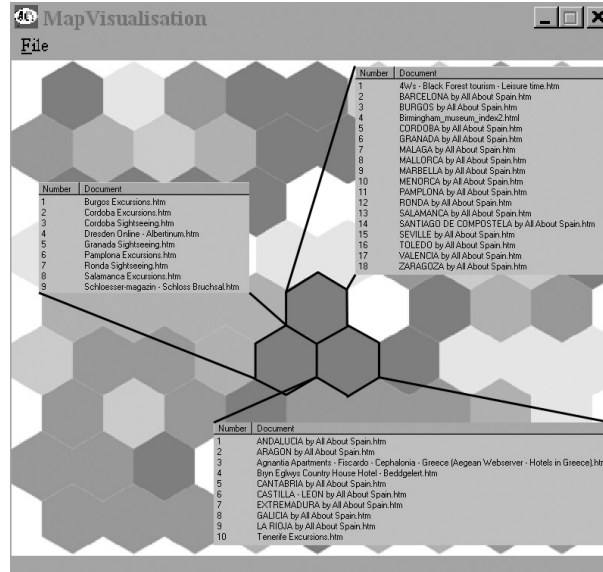


Fig. 3. The resulting document map DM for a 9×9 lattice.

	<i>Retrieved</i>	<i>Not-Retrieved</i>	
<i>Relevant</i>	<i>r</i>	<i>x</i>	$n_1 = r + x$
<i>Not-Relevant</i>	<i>y</i>	<i>z</i>	
	$n_2 = r + y$		

Table 1. Contingency table for evaluating retrieval.

modeling (ML-BBMs) and performed a significant reduction in mean squared error during the training phase of the SOM algorithm for the construction of the word category map. Furthermore, the variant leads to better retrieval results than the standard method with respect to the recall-precision curve. As a result, it appears to be a compelling competitor of the standard maximum likelihood approach.

References

- [1] Yates, R.B., Neto, B.R.: Modern Information Retrieval. ACM Press (1999)
- [2] Manning, D., Schütze, H.: Foundations of Statistical Natural Language Processing. Cambridge, MA: MIT Press (1999)
- [3] Goodman, J.T.: Putting it all together: Language model combination. Proc. of ICASSP'00 III (2000) 1647–1650
- [4] Jelinek, F., Mercer, R., Roukos, S.: Principles of lexical language modeling for speech recognition. Advances in Speech Processing. S. Furui and M.M. Sondhi, Eds., Marcel Dekker, New York (1992) 651–699

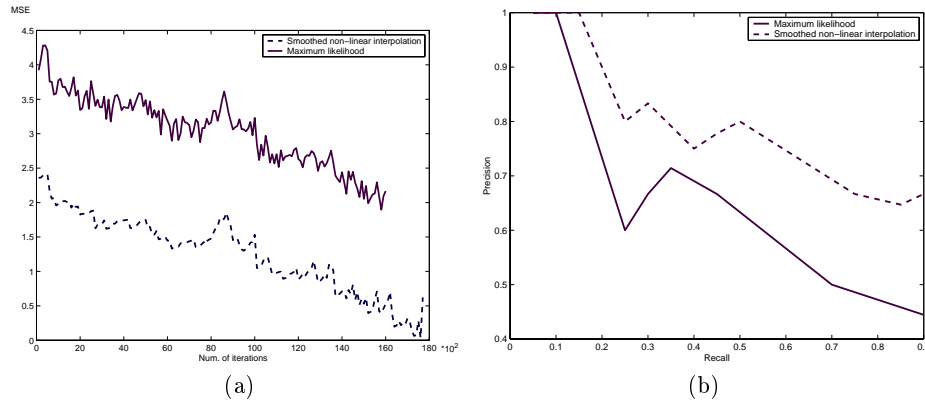


Fig. 4. (a) The mean squared error curve for the word category map of the 27×27 network for the NLI-BBMs and ML-BBMs models respectively; (b) the recall precision curve for the retrieval of relevant documents in the DM for the NLI-BBMs and ML-BBMs models, respectively.

- [5] Jelinek, F.: Statistical Methods for Speech Recognition. MA: MIT Press (1999)
- [6] Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. Proc. of SIGIR'98 (1998) 275–281
- [7] Brown, P.F., Della Pietra, V.J., De Souza, P.V., Lai, J.C., Mercer, R.L.: Class-based n-gram models of natural language. *Comp. Ling.* **18(4)** (1992) 467–479
- [8] Ney, H., Martin, S., Wessel, F.: Statistical language modeling using leaving-one-out. in *Corpus-Based Methods in Language and Speech Processing*. S. Young and G. Bloothoof, Eds., Kluwer Academic Publishers, Dordrecht, The Netherlands, (1997) 174–207
- [9] Becchetti, C., Ricotti, L.P.: *Speech Recognition: Theory and C++ Implementation*. New York: J. Wiley (1999)
- [10] Kaski, S., Lagus, K., Honkela, K., Kohonen, T.: Statistical aspects of the websom system in organizing document collections. in *Computing Science and Statistics*. D.W. Scott, Ed., Interface Foundation of North America, Inc., Fairfax Station, VA (1998) 281–290
- [11] Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Paatero, V., Saarela, A.: Organization of a massive document collection. *IEEE Trans. on Neural Networks.* **11(3)** (2000) 574–585
- [12] Kohonen, T.: *Self Organizing Maps*. Germany: Springer-Verlag (1997)
- [13] Kaski, S.: Dimensionality reduction by random mapping: Fast similarity computation for clustering. Proc. of IJCNN'98 **1** (1998) 413–418
- [14] Georgakis, A., Kotropoulos, C., Bassiou, N., Pitas, I.: Hypergeo: A data organization and retrieval system for tourist information. Proc. of IASTED Conf. on Applied Informatics (2001) 719–724
- [15] Frakes, W.B., Baeza-Yates, R.: *Information Retrieval: Data Structures and Algorithms*. Upper Saddle River: Prentice-Hall (1992)
- [16] Porter, M.F.: An algorithm for suffix stripping. *Program* **14(3)** (1980) 130–137
- [17] Korfhage, R.R.: *Information Storage and Retrieval*. New York: J. Wiley (1997)