Hebbian Principal Component Clustering for Information Retrieval on a Crowdsourcing Platform

Thomas Niederberger*, Norbert Stoop*, Markus Christen†, Thomas Ott*
* ZHAW Zurich University of Applied Sciences, Switzerland
Email: nith@zhaw.ch, stoo@zhaw.ch, ottt@zhaw.ch
†University of Zurich, Switzerland & University of Notre Dame, USA
Email: christen@ethik.uzh.ch

***Abstract*—Crowdsourcing, a distributed process that involves outsourcing tasks to a network of people, is increasingly used by companies for generating solutions to problems of various kinds. In this way, thousands of people contribute a large amount of text data that needs to already be structured during the process of idea generation in order to avoid repetitions and to maximize the solution space. This is a hard information retrieval problem as the texts are very short and have little predefined structure. We present a solution that involves three steps: text data preprocessing, clustering, and visualization. In this contribution, we focus on clustering and visualization by presenting a Hebbian network approach that is able to learn the principal components of the data while the data set is continuously growing in size. We compare our approach to standard clustering applications and demonstrate its superiority with respect to classification reliability on a real-world example.**

I. Introduction

Blogs, message boards, crowdsourcing forums, chatrooms, ... – the Web is full of platforms, where people share information. The content on these platforms is typically structured according to the temporal order of the entries, leading to topical threads. With a growing number of entries, users lose track of the information and repetitions with regard to content are likely to occur. In the case of crowdsourcing platforms, where contributors submit ideas or solutions to a posted problem and the initiator of a crowdsourcing process (crowdsourcer) awards the best contributions, repetitions are undesirable and the lack of a general overview hinders the exploration of the solution space. The crowdsourcer aims to gather a huge variety of ideas, spanning a wide solution space that maximizes the potential to find the best solution for the problem posed. As a crowdsourcing process may yield thousands of contributions, both the crowdsourcer and the contributor face the tasks of structuring, classifying and evaluating the contributions. These tasks should be performed in real-time, i.e. during the submission process.

This information retrieval problem involves the steps preprocessing, clustering and visualization. The basic methodology, as used for instance in internet search engines, starts with a vectorization of the texts and constructs a 'term by document' or 'term frequency-inverse document frequency' (TF-IDF) matrix[1][1], [2]. The matrix contains information about the occurrence of each (semantically relevant) term in a document, based on which, a proximity matrix for document clustering can be constructed. An alternative to a TF-IDF matrix is provided by probabilistic topic models [3] that are based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words [2], [3]. The task of information retrieval then amounts to inferring the model parameters from a given corpus (i.e. set) of texts, i.e. the goal is to find the best set of parameters that explain the observed data.

For the visualization of data from a document corpus, a variety of methods have been suggested whose commonality is the idea of dimensionality reduction. Among the most popular methods are multidimensional scaling (MDS), self-organizing maps (SOM), or principal component analysis (PCA). Depending on the characteristics of the data set, certain approaches are favoured over others [4].

Applying these standard methods to our problem is faced with various challenges: First, the texts are usually very short (ideas are described mostly by one to three sentences), lack formal structure, and are prone to errors, sloppy writing, and multilingualism (ideas may

[1]In this document we treat 'text' and 'document' as well as 'word' and 'term' as synonymous expressions.

contain words of different languages). This means that the information content per text is low and unstable (i.e. may change dramatically if a misspelled word is not recognized), which makes it difficult to compare and group the texts such that the latent topics of each group can be identified. Second, the number of topics is high (as maximizing the number of topics is one of the goals of the crowdsourcing process), which leads to a high dimensional feature space. This complicates the visualization task and consequently the identification of yet unexplored topics, as the results have to be presented in an easy-to-grasp visual form representing the basic structure of the underlying textual space. Traditional visualization techniques such as MDS or PCA are inadequate for this task, because they employ a 2-dimensional projection with a high information loss. Third, the process is *dynamic*, i.e. the number of texts and thus the vocabulary and the number of topics increases with time. The users need immediate, i.e. real-time, feedback requiring efficient algorithms. Nonetheless, the visualization should not display abrupt changes as they can be confusing to users and crowdsourcers. Rather, changes should happen continuously and almost imperceptibly. In this contribution, we introduce a novel information retrieval method that masters these challenges.

II. Methods

A. Preprocessing

For preprocessing, we use well known methods from natural language processing: The document corpus is converted to a bag of word model [3] describing the frequency of each word. A typical document corpus consists of 300 to 1000 texts with an average length of 54 words. Each text consists of a title, a number of tags and a description. The texts are usually in German without any additional information or structure. Non-German words are translated using standard dictionaries and orthographic errors are identified and corrected using standard spelling checkers. Then, all documents are transformed to lowercase. Technical sequences such as URLs are removed and country specific characters (e.g., German umlauts) are replaced by corresponding letters from the English alphabet. Each document is then split into a sequence of single words (1-gram model). Words without semantic information (stopwords) are removed from the sequence [6], while the semantic content is enriched by comparing the remaining words to an open-source synonym data set [7] and replacing synonyms by one word. Finally a German Porter stemmer [9] is applied in order to replace words by their word stem.

The result is used to calculate the TF-IDF matrix that consists of a term frequency part and a normalizing factor which reduces the importance of very frequently occurring terms (IDF part). The inverse document frequency for the term i is defined as the logarithm of the total number of documents N in the corpus divided by the number of documents containing the term n_i:

$$idf(i) = log(\frac{N}{n_i}) \tag{1}$$

The element v_{di} of the TF-IDF matrix is then defined as

$$v_{di} = tf(i,d) * idf(i) \tag{2}$$

where the term frequency $tf(i,d)$ denotes how often the term i occurs in document d. The resulting $N \times l$ matrix (where l stands for the number of terms) gives a description on how often a term occurs in a text. This matrix serves as input for the clustering algorithm.

B. Clustering and Visualization

Our clustering method is based on a Hebbian network approach that is able to learn the principal components of the data in a continuous way. Several papers in natural language processing already proposed using PCA. However, none of them considered the problem of a growing dataset in a real-time environment. For the visualization part, we had to find a way to embed several clusters, that are almost equidistant in two dimensions. We propose a ring-like structure for the rough structure, combined with a PCA for the fine structure. To break the visual impression of proximity in 2D we use a differentiated colouring of the clusters. Figure 1 illustrates the visualization part of our algorithm by using data emerging from a competition of the crowdsourcing platform www.atizo.com. In these competitions, companies request ideas for solving specific business problems. The example refers to a competition that generated 389 contributions (Fig. 1). In the following, we will first describe the basic idea of our algorithm and then we outline the role of Hebbian learning as a means of making the basic algorithm applicable in real-time.

Basic Algorithm: The algorithm is based on a PCA consisting of two parts: the first part (steps 1-3) clusters the dataset, the second part (steps 4-6) performs the visualization. The steps can be summarized as follows:

1) Run a PCA (using the data from the TF-IDF matrix) and reduce the feature space to k dimensions, where k is the number of clusters (for the choice of k see below)

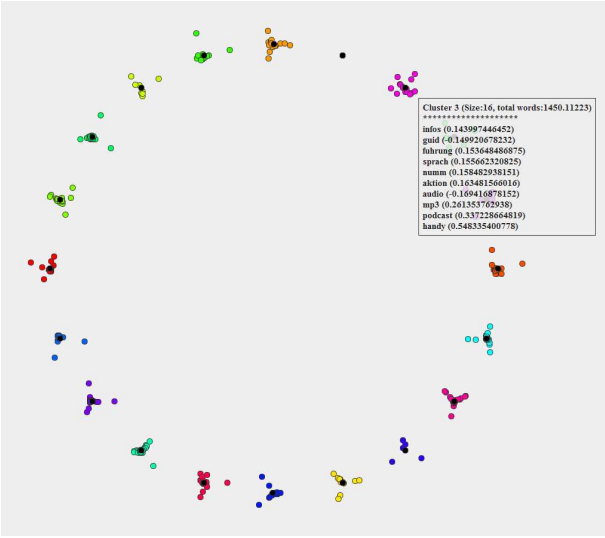

Fig. 1. Clustering result including a cluster description. A random colour was applied to all points of a cluster. Black points represent cluster centres.

2) Project all data items to the new k dimensional space.
3) Assign each data item to the principal component which has the biggest absolute coordinate value for this item.
4) Place the k clusters on the unit circle with evenly spaced arguments ranging from $0 - 2\pi$.
5) For each cluster perform a 2-dimensional PCA, using only the data assigned to this cluster.
6) Place all the data items according to their 2-dimensional components and move the origin of the coordinate system to the location of the cluster

This algorithm allows us to get a topical characterization of a cluster using the corresponding principal component: each principal component represents a topic consisting of a list of relevant terms. These relevant terms are found by determining the most important loadings of a principal component.

This basic algorithm has two drawbacks. First, it does not allow an easy integration of new data – neither of new features nor of new items – as the matrices must be recalculated in order to adapt to new data. Second, a single new data item is able to completely change the result space (e.g. changing direction of an axis) if standard PCA algorithms are used.

Inclusion of Hebbian Learning: To address these drawbacks we implemented a Hebbian learning variant of PCA in steps 1 and 5. Its solution is guaranteed to converge towards a standard PCA solution [5]. In the following, we only provide an outline of the generalized Hebbian algorithm (GHA), for further details see [5]. To start, we expect $N \geq 10$ for the learning procedure. The algorithm can be summarized as follows:

1) Initialize a neural network with l input neurons and k output neurons.
2) Initialize the synaptic weights w_{ji} to small random values with $j = 1, 2, ..., k$ and $i = 1, 2, ..., l$. Assign a small value to the learning rate $\eta < 1$ and set $n = 1$.
3) Choose an input vector x randomly from the data (corresponding to a row of the TF-IDF matrix).
4) Calculate the output according to
$$y_j(n) = \sum_{i=1}^{l} w_{ji}(n) x_i(n) \tag{3}$$
where $y_j(n)$ denotes the value of output neuron i and $x_i(n)$ the input to neuron i.
5) Calculate Δw following the Hebbian learning rule
$$\Delta w_{ji}(n) = \eta[y_j(n)x_i(n) - y_j(n) \sum_{k=1}^{j} w_{ki}(n) y_k(n)] \tag{4}$$
6) Adapt the synaptic weights w_{ji} according to
$$w_{ji}(n+1) = w_{ji}(n) + \Delta w_{ji}(n) \tag{5}$$
7) Set $n = n + 1$ and repeat steps $3 - 6$ until convergence.

After convergence, the synaptic weights of output neuron j represent the j-th principal component. Using GHA the system becomes adaptive to both increasing number of documents and increasing number of features. As soon as a new document introduces a new term to the document corpora, a new input neuron is being added to the network and therefore the network adapts to the new feature space.

Choice of parameter k: An open question is how to decide upon an optimal k. Various approaches such as minimum description length (MDL) [8] were proposed to decide upon this question. In our problem scope it seems reasonable that k should increase with an increasing number of texts. It should be noted, however, that for an optimal choice of k, not only machine learning concepts but also insights from human perception and cognition research have to be considered. Currently, k is chosen manually such that the results can be interpreted by a human being in a meaningful way. This is not a hard criterion and thus k can vary. For the example in Figure 1 we chose $k = 20$.

III. RESULTS

In order to evaluate the clustering performance of our algorithm on a benchmark dataset, we used four competitions from www.atizo.com. From each competition 50 contributions were randomly chosen and combined into a single dataset. The task of the algorithm was to identify the four clusters, i.e. to reconstruct the assignment of the contributions to the competitions. We compared our proposal with hierarchical Ward clustering and K-means. To improve the results, the feature space was reduced to 50 dimensions using a PCA before Ward clustering and K-means were applied. As K-means and hierarchical ward clustering showed similar results, we only report the results of K-means. In order to quantify the performance of the algorithms, we used confusion matrices. The results show that our approach is able to reconstruct all four competitions (Table I), whereas K-means and hierarchical clustering are only able to reconstruct one competition (Table II).

In addition, we compared both results to the reference clustering C_{ref} (defined by the competitions) using a Jaccard coefficient [10]. The Jaccard coefficient for two clusterings, C and C_{ref}, is defined as

$$J(C, C_{ref}) = \frac{a}{a+b+c} \tag{6}$$

where a is the number of pairs of items that are both in C and C_{ref} in same clusters, b is the number of pairs that are only in C in same clusters, and c is the number of pairs that are only in C_{ref} in same clusters. Clearly, $J(C, C_{ref})$ ranges between 0 and 1, with a value close to 1 signifying similar clusterings. For our method a Jaccard coefficient of 0.64 is achieved, whereas for K-means clustering the coefficient yields only 0.32, demonstrating the superiority of our approach.

	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Competition 1	48	0	1	1
Competition 2	0	45	2	3
Competition 3	0	6	36	8
Competition 4	0	0	4	46

TABLE I
CONFUSION MATRIX FOR HEBBIAN PRINCIPAL COMPONENT CLUSTERING

	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Competition 1	1	8	0	41
Competition 2	0	50	0	0
Competition 3	0	49	1	0
Competition 4	0	50	0	0

TABLE II
CONFUSION MATRIX FOR K-MEANS CLUSTERING

IV. CONCLUSION

We have presented a novel method for information retrieval adapted to a hard problem in text data analysis with low and instable information content of the single documents and a high dimensional and increasing feature space. The method outperforms standard clustering algorithms with respect to classification reliability, has the potential for real-time classification during a crowdsourcing process, and provides an innovative visualization for dealing with the problem of dimensionality reduction. In order to implement and validate the method for real-world applications, further steps are needed: First, the clustering has to be validated with respect to accuracy compared to manual clustering. Second, the method has to be integrated into the routines of the crowdsourcing web-interface and server infrastructure and prove its reliability and real-time capacity. Third, tests must demonstrate the positive effects of using our methods with respect to solution space exploration and improvement of the innovation process through crowdsourcing.

ACKNOWLEDGMENT

This project is supported by KTI grant 12747.1 PFES-ES. We thank our collaborators R. Hofstetter, S. Aryobsei, the team of S. Fabrikant, and the co-workers of Atizo.

REFERENCES

[1] D. Jurafsky, J. H. Martin. *Speech and language processing*. London (Prentice Hall), 2009.

[2] D. M. Blei, A. Y. Ng, M. I. Jordan. Latent Dirichlet Allocation. In *Journal of Machine Learning Research 3*, pp. 993–1022, 2003.

[3] M. Steyvers, T. Griffiths. Probabilistic Topic Models. In *Latent Semantic Analysis: A Road to Meaning*. (Lawrence Erlbaum), 2007.

[4] A. Skupin, S. I. Fabrikant. Spatialization Methods: A Cartographic Research Agenda for Non-geographic Information Visualization. In *Cartography and Geographic Information Science 30(2)*, pp. 95–115, 2003.

[5] S. Haykin. *Neural networks. A comprehensive foundation*. London (Prentice Hall), 1999.

[6] S. Bird, E. Loper, E. Klein. *Natural Language Processing with Python*. Sebastopol (O'Reilly Media Inc.), 2009.

[7] D. Naber. OpenThesaurus: ein offenes deutsches Wortnetz. In *Beiträge zur GLDV-Tagung*, pp. 422–433, 2005.

[8] J. Rissanen. Modeling By Shortest Data Description. In *Automatica 14*, pp. 465–471, 1978.

[9] M.F. Porter. An algorithm for suffix stripping, In *Program 14(3)*, pp. 130–147, 1980.

[10] T. Ott, A. Kern, A. Schuffenhauer, M. Popov, P. Acklin, E. Jacoby, R. Stoop. Sequential Superparamagnetic Clustering for unbiased Classification of High-Dimensional Chemical Data. In *J.Chem.Inf.Comput.Sci. 44*, pp. 1358–1364, 2004.