Self-Organizing Maps for Classification of a Multi-Labeled Corpus

Lars Bungum and **Björn Gambäck**
Department of Computer and Information Science
Norwegian University of Science and Technology
Sem Sælands vei 7–9, 7094 Trondheim, Norway
{larsbun,gamback}@idi.ntnu.no

Abstract

A Self-Organizing Map was used to classify the Reuters Corpus, by assigning a label to each of the documents that cluster to a specific node in the Self-Organizing Map. The predicted label is based on the most frequent label among the training documents attributed to that particular node. Experiments were carried out on different grid sizes (node numbers) to determine their influence on classification results. Informative visualizations of the resulting Self-Organizing Maps are demonstrated. We argue that the Self-Organizing Map is well suited to classify a document collection in which many documents simultaneously belong to several categories.

1 Introduction

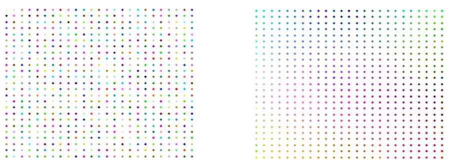
Categorization of a text corpus in which each article is attributed with a set of categories (labels), is a classical *supervised* classification task. Most supervised classification methods learn parameters from a training set of labeled instances, and use the learned model to score test instances. The Self-Organizing Map (SOM) is in contrast an *unsupervised* technique, clustering similar training instances together, without knowledge of their categories. The resulting maps display visually identifiable, but non-delimited, clusters. In this way, the SOM algorithm makes underlying similarities in high-dimensional space visible in lower dimensions. Through a two-step methodology, the labels of a training corpus associate with areas of the map; areas that in turn can be used to classify previously unseen documents.

The Reuters Corpus (Lewis et al., 2004) consists of text documents with a varying amount of labels attributed to them. Sebastiani (2002) notes a fundamental distinction between the *single-label* and *multi-label* classification tasks. In the former, only one label is attributed to each document, whereas any number can be attributed to the documents in the latter. Most research has gone into the single-label problem, as this will generalize to multi-label classification, by transforming the problem to a sequence of binary classification problems. This however, rests on the assumption that the categories are stochastically independent, that is, that the label of a document does not depend on the whether the document also has another label.

Another key problem in document classification relates to vectorization, how a group of documents is converted into a vector-based feature representation. The way documents are represented, and often the cut-off point in deriving TFIDF or ngram statistics, will result in different amounts of features. Documents can in principle be vectorized to any dimensionality, in its simplest form counts of occurrences of a given set of words.

A number of experiments were conducted on two different portions of the Reuters Corpus, the Top 10 and Full set of categories, respectively. While this follows a tradition of SOM-based classification, we offer more details on the implementation, the vectorization, and the parameters for creating the map. Self-Organizing Maps of different sizes were used to evaluate the importance of the number of nodes in SOM classification. The cost of computing the maps increase with the size of their topologies, posing the research question: Can this be justified with classifier performance?

We demonstrate that the method elucidates similarities between labels as well as between documents, and argue that the reduction of the multi-label classification of the corpus into a cascade of binary classification problems under the assumption that the categories are stochastically independent is not plausible.



(a) First epoch. (b) Last epoch.

Figure 1: RGB color coding of a SOM.

Section 2 discusses related work and presents some background on the SOM algorithm and using them as a classification device, while Section 3 describes the Reuters Corpus and Section 4 goes through the applied methodology. Section 5 presents the experimental results, that are further discussed in Section 6. Section 7 concludes and looks forward.

2 Self-Organizing Maps

Self-Organizing Maps (Kohonen, 1982) is a Cluster Analysis Algorithm with roots in Artificial Neural Networks, also termed Kohonen Neural Networks, as discussed by Lo et al. (1991). A SOM functions on nodes organized according to a given topology (usually 2-dimensional). The nodes are made up by vectors of some higher dimensionality, directly comparable to vectors representing training samples. The properties of the nodes are gradually changed during a pre-defined number of epochs, making the nodes more similar to the training samples. Changing one data point (node) will also affect its neighboring nodes, in a fashion inspired by biological systems in which neurons with similar functions organize in the same areas.

This process is illustrated in Figure 1, where 25x25 nodes are represented as vectors with three dimensions, each with values between 0 and 1. The training samples consist of a set of fixed colors, encoded as RGB (red, green, and blue) values. After the end of training, the grid has “self-organized“ into areas of similar colors. This is accomplished by comparing each training sample to each node, according to a *distance metric*, and drawing the closest node (“the winner”) and its neighborhood towards the training sample. The samples are processed serially, and this is repeated for *n* epochs, as outlined in Figure 2.

```
1. Initialize nodes in the 2-dimensional topology.
2.   (a) Sample all vectorized documents from the
         training material in succession.
     (b) Find the closest match in the node layout.
     (c) Update this matching node and its neigh-
         bors to be closer to the sample.
3. Reduce learning rate and/or neighborhood size.
4. Return to Step 2 until end of all epochs.
```

Figure 2: SOM algorithm.

Samples can be of any dimensionality; the higher the dimensionality, the more computationally costly will each comparison be. The resulting map will group (self-organize) similar high-dimensional vectors together, that can be visually inspected in the original topology, a low-dimensional space. Hence a feat of cartography in the landscape of documents is achieved, not only because documents are found in the same areas on the map, but also because the distances between nodes are visible with color shadings or contour plots. In this way differences in high-dimensional space become apparent in lower dimensions, creating a dimensionality reduction.

Hyötyniemi (1996) used a SOM to extract features for document representation, based on the clustering of character trigrams, arguing that this would account better for linguistic features.

Eyassu and Gambäck (2005) and Asker et al. (2009) used Self-Organizing Maps to classify Amharic news text. Categorized by experts, each document in the training corpus was associated with a query. A merged query and document matrix (i. e., the vector representation of the collection) was used for training the SOM. They report using many epochs (up to 20,000) and achieved classification accuracies in line with comparable methods such as Latent Semantic Indexing (Deerwester et al., 1990).

In order to use the SOM for a supervised classification task, consisting of a training and test corpus, it is necessary to establish a mechanism by which to ascribe a label to each of the test samples, i. e., what class membership (label) to *predict* for the sample in question. During classification, a test sample will be compared to the 2-dimensional grid to find its *winning node*, just like in training.

Going from a winning node to a prediction requires attribution of labels to individual nodes. One way to choose the label to be attributed by a node is to assign the test sample all categories found in the node. Saarikoski (2009) and Saarikoski et al. (2011) suggested another straightforward method by which the label prediction is taken as the most frequent label, i. e., is selected

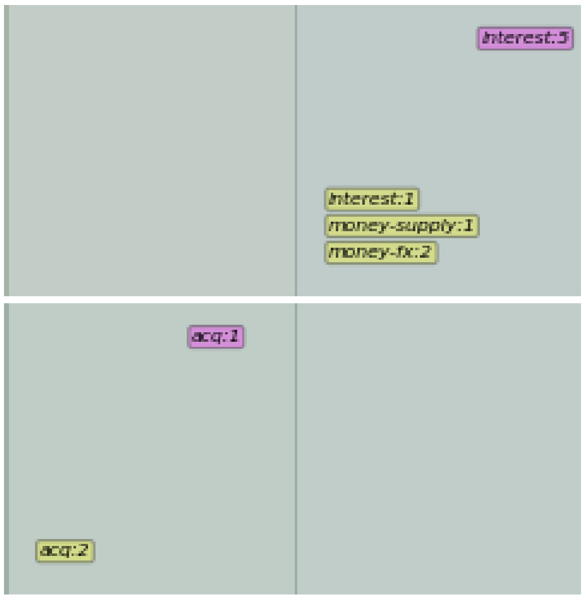


Figure 3: Example of SOM clustering with majority voting. Yellow labels from training corpus, magenta labels from test corpus.

by majority vote of the class labels of all training samples that had this node as winner. In the last run of the SOM, the Best-Matching Unit (BMU) of each training sample in the final epoch is recorded.

For empty nodes, being no sample's BMU, the next closest n nodes (the BMUs being the closest) are investigated for training samples until a non-empty node is found. The prediction is made according to the most frequent label of this node.

This process is illustrated with Figure 3, where each square represents a node (so a 2x2 SOM). Two of the four nodes have documents attaching to them. The yellow labels are from the training corpus, and the magenta from test samples. The SOM classifies test documents correctly, if their labels match that attributed to them by the node.

In the bottom-left node, both matching training samples (2 in yellow) had the label *acq* (acquisitions), which would be the prediction of this node. The one test sample (1 in magenta) that had this node as its BMU was also labeled *acq*, i. e., a correct prediction.

For the top-right node, the figure shows that the four training samples were split between *money-fx* (2), *money-supply* (1) and *interest* (1). By majority voting, the predicted label of the node becomes *money-fx*. However, the five test samples were all labeled *interest*, and all of them would thus be falsely classified (as *money-fx*).

3 Data

The Reuters Corpus (Lewis et al., 2004) has been used extensively for text classification research. For easier comparison of results, several ways of processing the corpus have been established. Those pertain to the corpus subsets, and a specific split of it into training and test sets called the `ModApté` split (Lewis, 1997).

A further much used split means dividing the `ModApté` into either the ten most frequent categories, the categories with at least one positive training and one test example (90), or the categories with at least one training example (115).

The split with the 90 categories with at least one positive training and test example is somewhat confusingly called `AptéMod` by Yang and Liu (1999), while Debole and Sebastiani (2004) refer to it as the *R(90)* subset of the `ModApté` split. Elsewhere, the R(90) split with 10,789 documents is also called `ModApté` (Yang et al., 2009; Chen et al., 2004; Saarikoski et al., 2011).

The Natural Language Toolkit (NLTK) interface (Loper and Bird, 2002) to the Reuters Corpus provides the `AptéMod` split as comprised by 7,769 training examples and 3,019 test instances. The category frequencies are $[9160, 1173, 255, 91, 52, 27, 9, 7, 5, 3, 2, 1, 0, 2, 1]$, i. e., 9,160 documents with just one category, 1,173 with two, etc. A plot of the log-frequency of categories is shown in Figure 4.

The NLTK interface provides a data structure where the raw text and categories are indexed on filenames. This was used in the present work, and the documents were transformed into a matrix of TF-IDF values, i. e., the product of the Term Frequency (the occurrence of the term in the document) and the Inverse Document Frequency (the log of the number of documents containing this term). A term can be any n-gram.

The corpus is comprised of text documents with 0 to 15 different category labels. The five most frequent labels of the 90 categories in the `ModApté` split have the following counts: $[3923, 2292, 374, 326, 309]$, totalling 7,224 of the 9,160 documents. The skewed distribution means that accuracy on the entire corpus is not always a good measure of classifier performance, as the performance on less frequent categories may drown in the larger categories. The distinction between micro- and macroaveraging mitigates this.

The *microaverage* averages over the categories

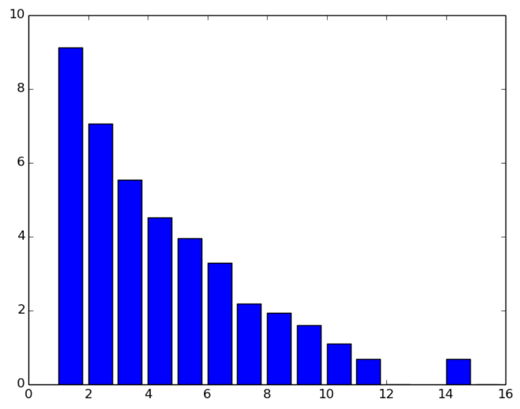


Figure 4: Log-frequencies of categories in the Reuters Corpus. Log of 0 set to 0.

to be classified within an experiment counting all correct predictions in one pool and dividing by the number of classified documents. This is expressed in Equation 1 where c denotes the number of correctly classified documents and n the total number of documents.

$$Microaverage = \frac{c}{n} \tag{1}$$

This measure will be skewed towards larger classes: Consider a classifier that classifies one large category with 90 documents 100% correctly, whereas 10 other classes with one document each were all wrong. That would give a microaverage of 90%, even though most categories were completely wrong.

The *macroaverage* is a measure that is weighted with relative size, expressed formally in Equation 2 where c_j and n_j are the number of correctly classified documents belonging to class j and the total number of documents in that class, resp.

$$Macroaverage = \frac{\sum \frac{c_j}{n_j}}{|Classes|} \tag{2}$$

In the same way as for accuracy, micro- and macroaverages can be calculated for precision, recall and F-score. However, when the classification problem is framed as a *one-of* classification, i. e., when all documents in the test set belong to exactly one class, the microaveraged F-score will be the same as the accuracy, because the number of false positives and false negatives will always be the same. If a document is classified with a label it does not have, it will be a false positive in that class, but also a false negative in the class it correctly belongs to (Manning et al., 2008).

4 Implementation

The Self-Organizing Map algorithm was implemented in MPI (mpi4py)[1] and Python, and run on a Portable Batch System (PBS)[2] scheduler. A quadratic layout of nodes was used for the experiments. Samples were processed serially, but for each sample, the vector comparisons and updating of nodes were done in parallel. The availability of large-scale High-Performance Computing (HPC), facilitated feasibility of the "on-line" version of the SOM algorithm. "On-line" as weights are updated after processing of each training sample, as formulated in Figure 2.

Methods for reducing computational cost include using a two-step approach (i. e., a SOM on the output of another SOM) (Kohonen et al., 1996), or formulating a "batch-SOM" algorithm. These methods contrast the "on-line" version by updating all node weights in one operation per epoch. See Lawrence et al. (1999) and Patel et al. (2015) for parallel batch-SOM implementations. Fort et al. (2002) compared the approaches and noted some problems with the batch formulation, e. g., initialization sensitivity, while it had advantages in terms of speed and efficiency.

In the on-line formulation, vector comparisons and updates are run per training sample, increasing linearly with the number of training samples. In turn, the vector comparisons increase with the number of nodes in the grid. Vector comparisons are costly, and therefore lend themselves well to parallelization. Hence, the parallel on-line implementation used in these experiments is sensitive to a large amount of training samples, but well equipped to handle large SOM topologies.

Each experiment in Section 5 was defined in a configuration file, where parameters such as the size of the grid, the number of iterations, the learning rate and the size of the initial neighborhood radius were specified, as well as details about the vectorization of training data. The neighborhood surrounding each BMU was defined by a diminishing-by-epoch radius, with a configurable initial size. The radius was reduced by exponential decay, as was the learning rate, i. e., the degree to which Best-Matching Units were updated to be similar to training samples.

The vectorization of documents was done with scikit-learn (Pedregosa et al., 2011), offering

[1]`http://mpi4py.scipy.org/`

[2]Both OpenPBS and PBS Professional were used.

a large selection of convenient vectorizers that swiftly transform test documents into the same vector form as the training corpus. In these experiments, the TFIDF transformer was used for vectorization. Similarly the library offers a number of metrics for vector comparison, such as Euclidean, Hamming or Chebichev distances. An Euclidean distance metric was used for all the following experiments. The implementation was fully modular with regard to the choice of these methods.

5 Experiments

Two rounds of experiments were carried out. The first experiments were conducted with the same vectorization parameters on five different grid sizes; 8x8, 16x16, 32x32, 64x64, and 128x128 (i.e., from 64 to 16,384 nodes) on the Top 10 categories of the Reuters Corpus, comparing execution times for grid configurations with increasing numbers of parallel processes. The second round of experiments was done on the entire `AptéMod` split (90 categories), investigating the effect on classification performance of varying grid sizes by the same amount. In both rounds of experiments, the SOM was configured with an initial learning rate of 0.10 and an initial neighborhood radius of a quarter of the grid dimension.

In order to focus experiments on the *one-of* classification problem, the training and test corpora were limited to documents with only one category when classifying the Top 10 categories. When using majority voting for prediction, the method would not benefit from the added information that some documents have more classes, as less frequent classes in each node could be voted down. Using documents restricted to one label could therefore bring about better separation in the Self-Organizing Map.

5.1 Top 10 Categories

In these experiments, the documents were limited to those belonging to the Top 10 categories. Documents were vectorized into 33,120 dimensions. Each vector consists of the TFIDF values for the ngrams ranging from 1 to 7, with a cut-off frequency of 5 (i. e., ignoring dictionary terms with a frequency below this threshold). The number of documents labeled with these categories are listed in Table 1.

The first experiments are summarized in Table 2. While we did not exhaustively experiment with the optimal number of parallel processes for all experiments — i. e., finding the sweet spot beyond which new processes do not mean a speed-up due to overhead costs — we did one experiment on creating the same SOM (identical parameters) with a different amount of nodes processes in use on the High-Performance Computing (HPC) grid. The experiment compared creating a SOM with a 16x16 node grid, computed with 64 processes over 4 nodes (the same as in Table 2), to running it all on just one node, with parallel 16 processes. The former (64 parallel processes) took 1 hour to complete vs. 4 hours when running on only one node (16 processes).

Category name	Num. train	Num test
earn	2840	1083
acq	1596	696
money-fx	222	87
crude	253	121
trade	250	76
interest	191	81
ship	108	36
sugar	97	25
coffee	90	22
grain	41	10

Table 1: Number of training/test samples in Top 10 categories of the AptéMod restricted to documents with only one category.

Grid size	8x8	16x16	32x32	64x64	128x128
Microavg.	0.80	0.89	**0.89**	0.85	0.80
Macroavg.	0.52	0.79	**0.82**	0.78	0.75
Processes	8	64	256	1024	2048
Hours	0.5	1	2	4.5	11.5

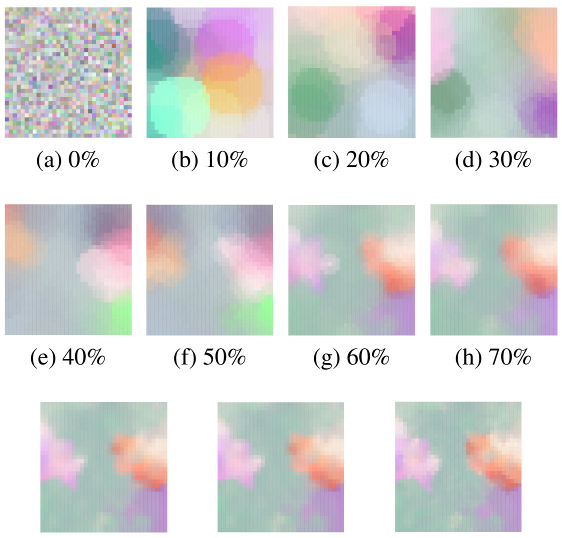
Table 2: Classification scores for Reuters Corpus (Top 10 categories) across grid sizes.

5.2 All Categories

The second group of experiments was conducted on all (90) categories, summarized in Table 3.

Grid size	8x8	16x16	32x32	64x64	128x128
Microavg.	0.65	0.75	**0.78**	0.77	0.76
Macroavg.	0.07	0.13	0.25	0.25	**0.30**

Table 3: Classification scores for Reuters Corpus (all categories).



(a) 0% (b) 10% (c) 20% (d) 30%

(e) 40% (f) 50% (g) 60% (h) 70%

(i) 80% (j) 90% (k) 100%

Figure 5: Development of SOM Top 10 categories vectorized with 33,120 dimensions, 32x32 grid and 100 iterations.

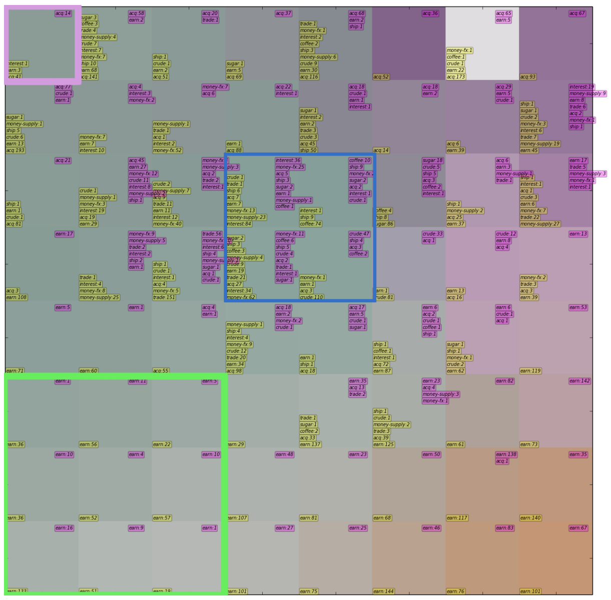


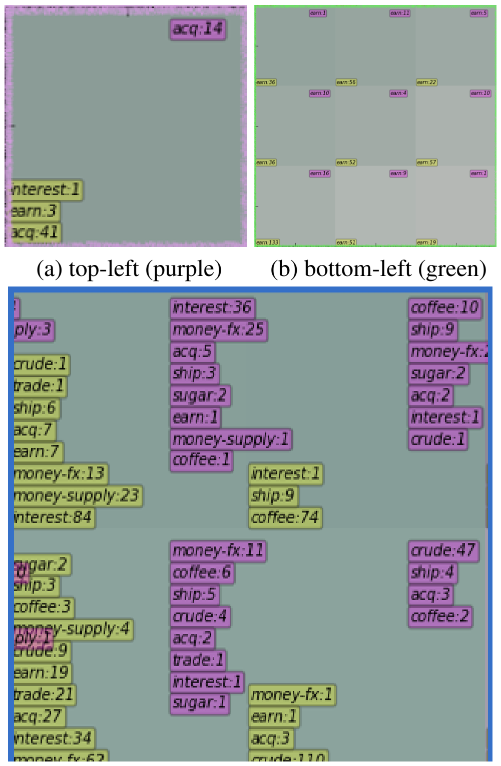
Figure 6: Annotated SOM with grid size 8x8 of the Top 10 categories, with highlighted excerpts. Training samples yellow, test samples magenta.

In these experiments, documents were vectorized into 19,092 dimensions. Each vector was comprised by TF-IDF values for ngrams ranging from 1 to 7, with a cut-off frequency of 10. While documents both in training and test sets could have multiple labels, each label was treated individually when predicting and scoring, as a series of single-label classification problems.

5.3 Visualization of SOM Development

Figure 5 shows the development of a Self-Organizing Map with a grid size of 32x32 from the experiments on the Top 10 categories. During the first iteration steps, the radius of the initial neighborhood is clearly visible before the grid self-organizes into a map. Each node is represented by a vector with the same dimensionality as the vectorized documents; these node vectors were reduced to four dimensions using Principal Component Analysis (PCA) (Jolliffe, 1986).

The four values were then used to encode a color in the Matplotlib library (Hunter, 2007), displaying each node as a color grading. The library accepts vectors of both three and four dimensions to create colors, so in order to retain more of the variance in the original node vectors, the vectors were reduced to four dimensions. Similarity in colors visualize similarity between vectors in the same area. Document labels verify that these areas represent texts belonging to the same categories.



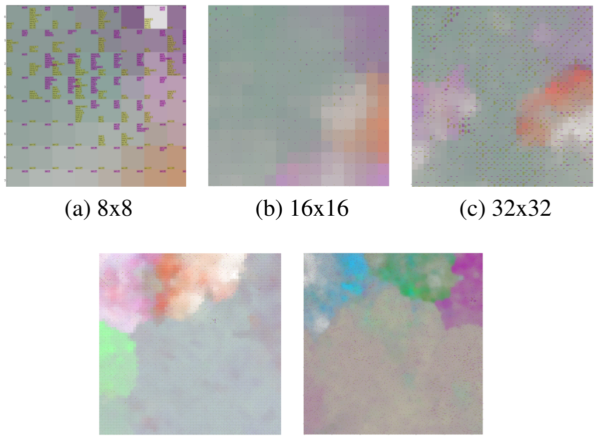
(a) top-left (purple) (b) bottom-left (green)

(c) middle (blue)

Figure 7: Excerpts from Figure 6 showing how document categories cluster.

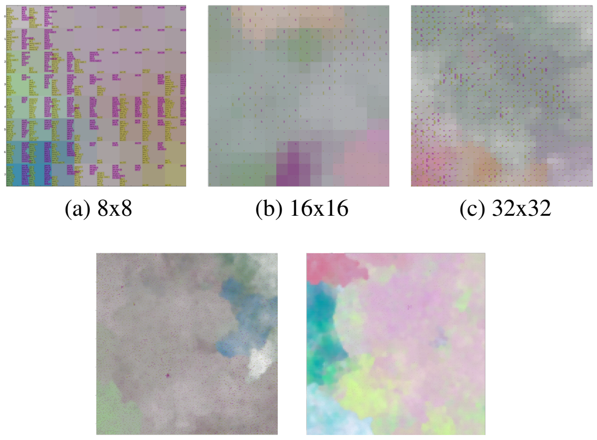
5.4 Visualization of Resulting SOMs

Figure 6 shows a completed Self-Organizing Map annotated with the categories of the training (yel-


(a) 8x8 (b) 16x16 (c) 32x32

(d) 64x64 (e) 128x128

Figure 8: Various size SOMs, Top 10 categories.


(a) 8x8 (b) 16x16 (c) 32x32

(d) 64x64 (e) 128x128

Figure 9: Various size SOMs, all categories.

low) and the test (magenta) samples.

The 8x8 map is used for illustration here since it is easier to show the contents of grid cells in this format. As the number of nodes grows larger, it is difficult to fit the areas in one frame.

Three excerpts of the 8x8 SOM of Figure 6 are detailed in Figure 7. Starting with Figure 7a, the cell in the top left corner of the map has three training corpus labels attaching to it, *acq* (41 documents), *earn* (3) and *interest* (1); and 14 documents from the test corpus, all with the category *acq*. Majority voting among the training samples assigns the category *acq* to the node and hence ensures that all of the test samples are labeled correctly.

In Figure 7b, all documents — both from the training and test corpora — have the label *earn*, so classification is straight-forward (and correct).

Finally, the four nodes in Figure 7c have attracted documents with multiple labels. The figure shows four cells, each representing a node in the SOM. The yellow training labels are sorted in decreasing order from the left bottom up, and the magenta test labels in decreasing order from the left top corner of the cell.

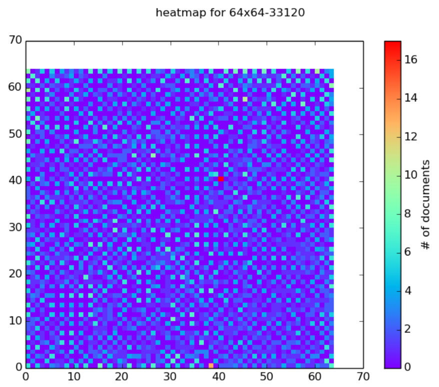
The labels with the highest frequency both in the training and test corpora match all four cells in Figure 7c (*interest*, *coffee*, *money-fx* and *crude*). Thus, these labels from the training corpus are the predictions for each node, and the set of correctly predicted test samples, respectively.

In addition to the most frequent labels, the lists in Figure 7c share many other members, indicating that the clustered documents have more properties in common.

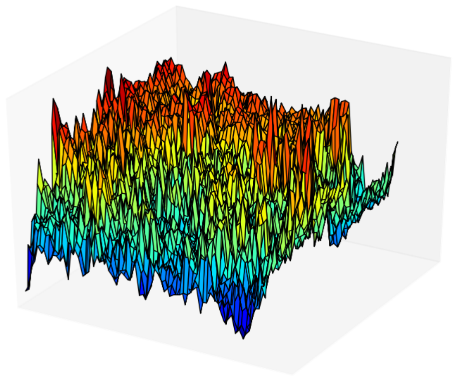
The final maps for the Top 10 category and all category runs are shown in Figures 8 and 9. In the larger maps there are fewer categories attaching to each node, both from training and test corpus. As can be seen from the results in Tables 2 and 3, macroaveraged F-scores increase as the grid sizes increase. With fewer categories per node, smaller categories have a better chance of being the majority category of a node, enabling the SOM to predict that node.

Figure 8d shows a SOM with grid size 64x64. On the left side of the picture, a light green area is visible. The green area is populated by documents labeled *crude*, going into an area labeled *earn* as the area turns gray in the vertical direction. Likewise, in the middle of the figure, areas populated by *interest* neighbor areas labeled *money-supply*. The map images are too large to include, and hence left out from the present paper.

Figure 10 complements the information of the color shadings. The heatmap shows the amount of documents attached to each node (having the node as their BMU). This number is not visible from the color gradings of the map Visualization, and offers insight into the relative size of these areas. The Unified Distance Matrix (UDM) shows the differences between the nodes on the map. The UDM is computed by taking the average distance between each node and its immediate neighbors in 2D space. High distances between nodes are shown as peaks in Figure 10b, and low areas represent clusters of similar vectors. Comparing Figures 8d and 10a, the flat area on the UDM reflects the large area in the bottom right corner in the same gray shading.



(a) Heatmap



(b) Unified Distance Matrix

Figure 10: Heatmap and Unified Distance Matrix of SOM with grid size 64x64. Top 10 categories.

6 Discussion and Related Work

Using the unsupervised Self-Organizing Map approach we observe relations between categories, both through clusters of documents belonging to similar categories being placed next to each other on the map, and also in terms of some nodes in the SOM attracting separate documents with thematically adjacent labels as their Best-Matching Unit, such as *money-fx* and *interest*. This could be a reflection of these documents being in a thematical intersection between topics, or outright belonging to both.

Wermter and Hung (2002) integrated the SOM with WordNet-based (Miller, 1995) semantic networks for doing classification on the Reuters Corpus. Documents were represented by *significance vectors*, i. e., the degree to which the documents belong to certain preassigned topics, that were in turn calculated from the importance of words in each category.

Saarikoski (2009) performed several experiments on using Self-Organizing Maps for Information Retrieval and document classification (Saarikoski et al., 2011), comparing the algorithm to other machine learning techniques. While they explained how the SOM was applied to the problem, some factors that affect classification performance were unaccounted for, such as the parameters for the SOM creation, notable grid size and learning rate. Restricting the classification to the Top 10 categories (by frequency) their best experiments had micro- and macroaverages of 93.2% and 83.5%, respectively.

Interestingly, Saarikoski et al. (2011) report that a Naïve Bayes classifier outscored all other methods on the data (95.2% and 90.4%), results that also significantly beat the findings of Dumais et al. (1998), who reported only 81.5% accuracy for the Naïve Bayes method. This discrepancy is likely due to a difference in scoring, as Dumais et al. (1998) reported break-even scores (motivated by comparability with other research), as opposed to Saarikoski et al. (2011) that gave true accuracy scores without any adjustment for precision-recall trade-off. The break-even point is where the precision is equal to recall, the point at which false positive and negative mis-classifications are done at the same rate.

Saarikoski et al. (2011) noticed how the costly training of SOMs is followed by cheap testing of new instances when doing classification, and suggested using multiple maps for multi-label classification, or alternatively using the three nearest labels for a new instance. This would, however, assume that all documents should have the same amount of labels in multi-label classification, which clearly is not the case in general. Still, it is possible to equip each node with a notion of label distribution, among its BMU training samples, possibly within a region (neighborhood) of the BMU.

While we did not see an increase in the *one-of* classification when the Self-Organizing Map topologies went above 32x32 nodes, it is likely that the multi-label problem benefits from a larger amount of nodes, offering finer granularity.

7 Conclusion and Future Work

In this paper we have used a parallel implementation of an "on-line" Self-Organizing Map algorithm on a well-researched classification task. Expanding the analysis offered in comparable research, we have explored how the grid size of the

SOM affects classification performance. We observe that classification performance increases up to the size of 32x32 nodes, and then deteriorates for the vectorizations used in the experiments. The paper offers extensive details on the parameters used to create the SOMs.

Using the Self-Organizing Map analysis, we observe underlying relations between documents, visible in 2D space, although represented with high-dimensional vectors. In the Self-Organizing Maps, similarities between *labels* as well as similarities between individual documents are visible. We therefore argue that the simplification of the *multi-label* classification task to a cascade of binary classification tasks is not plausible for the Reuters Corpus, because of the presence of relations between labels.

Further research into the multi-label problem and its relation to large-topology Self-Organizing Maps is prudent, given the findings of the present work. Devising a fair evaluation scheme for measuring the accuracy of SOM classifiers that attribute many labels to each document instance as a multi-label problem, is still an open question. Similarly, it is also an open question which method to use to decide how many labels to attribute to a new training instance. It is a natural next step to run more experiments on attributing several labels to documents in one operation and comparing that to running a series of binary classifications, as in the experiments presented in this paper.

Having implemented the Self-Organizing Map algorithm in a highly modular fashion, we would also like to experiment with both a) using different vectorizers and b) applying different distances metrics, in order to investigate their impact on classification results. Wermter and Hung (2002) reported good results on integrating other knowledge sources in vectorization, calling for a systematic comparison to purely data-driven vector transformers, and to hybrids between knowledge- and data-driven vectorizers.

Acknowledgments

The research leading to these results has been funded by Norwegian University of Science and Technology (NTNU), and by the Norwegian Research Council (NFR) and the Czech Ministry of Education, Youth and Sports (MŠMT) through the Norwegian Financial Mechanism 2009–2014 under Contract no. MSMT-28477/2014.

References

Lars Asker, Atelach Alemu Argaw, Björn Gambäck, Samuel Eyassu Asfeha, and Lemma Nigussie Habte. 2009. Classifying Amharic webnews. *Information Retrieval*, 12(3):416–435.

Junli Chen, Xuezhong Zhou, and Zhaohui Wu. 2004. A multi-label Chinese text categorization system based on boosting algorithm. In *Proceedings of the Fourth International Conference on Computer and Information Technology*, pages 1153–1158, Washington, DC, USA. IEEE Computer Society.

Franca Debole and Fabrizio Sebastiani. 2004. An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, 56:971–974.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Susan Dumais, John Platt, Mehran Sahami, and David Heckerman. 1998. Inductive learning algorithms and representations for text categorization. In *Proceedings of the 7th International Conference on Information and Knowledge Management*, pages 148–155. ACM Press.

Samuel Eyassu and Björn Gambäck. 2005. Classifying Amharic news text using Self-Organizing Maps. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, Michigan, June. ACL. Workshop on Computational Approaches to Semitic Languages.

Jean-Claude Fort, Patrick Letrémy, and Marie Cottrell. 2002. Advantages and drawbacks of the Batch Kohonen algorithm. In Michel Verleysen, editor, *Proceedings of the 10th European Symposium on Artificial Neural Networks*, pages 223–230, Bruges, Belgium.

John D. Hunter. 2007. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

Heikki Hyötyniemi. 1996. Text document classification with self-organizing maps. In J. Alander, T. Honkela, and M. Jakobsson, editors, *Proceedings of 7th Finnish Artificial Intelligence Conference*, pages 64–72, Vaasa, Finland.

Ian T. Jolliffe. 1986. *Principal Component Analysis*. Springer Verlag, New York, NY, USA.

Teuvo Kohonen, Samuel Kaski, Krista Lagus, and Timo Honkela. 1996. Very large two-level SOM for the browsing of newsgroups. In C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff, editors, *Proceedings of ICANN96, International Conference on Artificial Neural Networks*, Lecture Notes in Computer Science, vol. 1112, pages 269–274. Springer, Berlin, July.

Teuvo Kohonen. 1982. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69.

Richard D. Lawrence, George S. Almasi, and Holly E. Rushmeier. 1999. A scalable parallel algorithm for self-organizing maps with applications to sparse data mining problems. *Data Mining and Knowledge Discovery*, 3(2):171–195.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, December.

David D. Lewis. 1997. Reuters-21578 text categorization test collection. `http://www.daviddlewis.com/resources/testcollections/reuters21578/`.

Zhen-Ping Lo, Masahiro Fujita, and Behnam Bavarian. 1991. Analysis of neighborhood interaction in Kohonen neural networks. In *Proceedings of the Fifth International Parallel Processing Symposium*, pages 246–249. IEEE.

Edward Loper and Steven Bird. 2002. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, November.

Bhavik Patel, Anurag Jajoo, Yash Tibrewal, and Amit Joshi. 2015. An efficient parallel algorithm for self-organizing maps using MPI-OpenMP based cluster. *International Journal of Computer Applications*, IJCA Proceedings on International Conference on Advanced Computing and Communication Techniques for High Performance Applications(2):5–9, February.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(1):2825–2830.

Jyri Saarikoski, Jorma Laurikkala, Kalervo Järvelin, and Martti Juhola. 2011. Self-organising maps in document classification: A comparison with six machine learning methods. In Andrej Dobnikar, Uroš Lotric, and Branko Šter, editors, *Adaptive and Natural Computing Algorithms: 10th International Conference, ICANNGA 2011, Ljubljana, Slovenia, April 14-16, 2011, Proceedings, Part I*, volume 6593 of *Lecture Notes in Computer Science*, pages 260–269. Springer.

Jyri Saarikoski. 2009. A study on the use of self-organised maps in information retrieval. *Journal of Documentation*, 65(2):304–322.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, March.

Stefan Wermter and Chihli Hung. 2002. Self-organizing classification on the Reuters news corpus. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yiming Yang and Xin Liu. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 42–49, New York, NY, USA. ACM.

Shuang-Hong Yang, Hongyuan Zha, and Bao-Gang Hu. 2009. Dirichlet-Bernoulli alignment: A generative model for multi-class multi-label multi-instance corpora. In Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2143–2150. Curran Associates, Inc.