HIERARCHICAL DOCUMENT CATEGORIZATION USING ASSOCIATIVE NETWORKS

Niels Bloom
Perrit B.V.
Hengelo, The Netherlands
email: n.bloom@perrit.nl

Mariët Theune
Human Media Interaction
University of Twente
Enschede, The Netherlands
email: M.Theune@utwente.nl

Franciska de Jong
Human media Interaction
University of Twente
Enschede, The Netherlands
email: f.m.g.dejong@utwente.nl

ABSTRACT
Associative networks are a connectionist language model with the ability to handle dynamic data. We used two associative networks to categorize random sets of related Wikipedia articles with only their raw text as input. We then compared the resulting categorization to a gold standard: the manual categorization by Wikipedia authors and used a neural network as a baseline. We also determined a human rating by having a group of judges rank the four categorization methods by correctness and by usefulness with regards to finding information. Based on these tests, we determined that associative networks produce results that are clearly better than the neural network baseline, coming close to the gold standard in terms of usefulness and correctness. Furthermore, automated testing suggests these results continue to hold for large datasets.

KEY WORDS
Associative Networks, Automatic Categorization, Connectionist Language Model, Document Clustering, Hierarchical Categorization

1 Introduction

Companies often build up large libraries of documents related to their field. To better share this knowledge, either amongst employees or directly with the customer, it is necessary to provide a hierarchical categorization to enable people to find the information they need.

Since the document collections can be large and dynamic, for example wikis where anyone may add, remove or edit any document at any time, hierarchical document categorization is an increasingly relevant and complex problem. The goal of a system tackling that problem is to automatically group the documents in a library into clusters and name and nest those clusters to form a hierarchical categorization. That categorization should match the content in an intuitive way, that is, both easily understood and sensible to human users. The system should also be able to deal with adding, removing and editing documents.

In earlier work [1], we used associative networks to classify documents into predefined classes. In this paper, we describe a new method of activating associative networks and extend our method to categorize rather than merely classify documents: the system determines its own hierarchical structure of categories to order the documents instead of relying on predefined classes as we did before.

We show that with associative networks we are able to draw up an intuitive categorization for sets of related Wikipedia documents that comes close to manual categorization in terms of usefulness and correctness, easily outperforming a neural network baseline and additionally prove that associative networks can do this in real time, allowing it to maintaining the categorization in a live environment, for both small and large datasets.

Below in Section 2 we describe our general approach to text categorization using associative networks, while in Section 3 we describe the associative network used in our experiments. Section 4 explains the setup of our experiments and Section 5 presents the results. Section 6 discusses related work. We end with conclusions and directions for future work in Section 7.

2 General Approach

An associative network is a connectionist approach used to mimic thought patterns to resolve problems that have no simple mathematical solution. Like neural networks [2, 3], which we use as a baseline in our experiments, it consists of a set of connected nodes with weights assigned to the connections. In this section we show how an associative network can be used to find connections between documents by taking one document as input, spreading activation or flow from it through the network and finding which of the other documents receives the most activation or flow.

2.1 Associative Networks

Associative networks are modelled after the associative thinking used by humans to solve certain problems [4, 5, 6]. The basis of the associative network is a connectionist model [7] in which each observation has its own node in the network. The nodes are connected by edges, modelling the associations between them. Figure 1 shows a simplified associative network consisting of five nodes with four connections.

In our approach to text categorization, individual

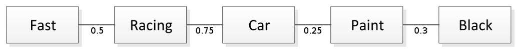

Figure 1. Simplified Associative Network

words from a text document are used as the basic observations modelled by an associative network. Each word is associated with other words in the network, for example the word *car* may be associated with the words *wheels* and *racing*. When a certain word is observed, other words are automatically activated if they are associated with that word. This can provide us with additional information about the words that is not immediately apparent. I.e., it allows us to create a list of words that may be related to the text, even though they are not in the document themselves.

As an illustration, let's look at the sentence *The fast black car was winning*. Using the association model, we are able to extract background information that is not explicitly provided. For example, both the word *fast* and the word *car* may be associated with racing. The word *winning* is also associated with racing. Thus, a document containing the sentence *The fast black car was winning* may be assigned to the category *racing* with other documents concerning this topic, even if it does not mention the word racing at all. This feature is especially useful in corporate libraries where a document targeting technical staff will use different language than a document intended for the uninformed customer, despite covering the same topic.

Background information found through association is not always perfect. Association allows us to make an educated guess about information rather than providing absolute certainty; the additional background information is inferred, not deduced. However, the more words in the text we can link with a concept (such as *racing*), the higher the probability that this concept is adequate.

Because the technique is applied to a very large problem space (language), the associative networks are generally large as well, numbering anywhere from hundreds of thousands to tens of millions of nodes. The number of connections between nodes, by comparison, is relatively small (up to a thousand edges per node), meaning the network forms a sparse graph.

2.2 Activating the Network

Once a network has been created (see Section 3.1 for an explanation of how we do this), it can be used to make associations between documents. To do so, it is activated by a certain input – typically a document, as a set of one or more words. For example, in Figure 2, the input value is 5 for *fast* and 4 for *car*, indicated by the incoming arrows and the marking of the nodes. The activation is spread from this input, activating neighbouring nodes in the network, which may in turn activate even more nodes. Nodes are activated

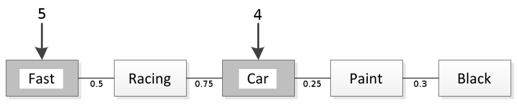

Figure 2. Input

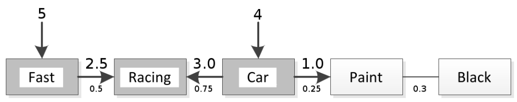

Figure 3. Activating

to different degrees depending on their distance to the input and the number and weight of the connections. If the activation falls below a threshold value, the node is not activated.

In Figure 3 the activation of the input is spreading towards the nodes *paint* and *racing*. Since *racing* is connected to *fast* with a factor 0.5 and *car* with a factor of 0.75, its activation value is 5.5. The activation value for *paint* which is connected only to *car* by a factor of 0.25 becomes 1. In this case, the threshold value of the *paint* node is greater than this (not indicated in the figure). Thus the node is not activated and not marked in the figure. Note that in an actual network the closer two concepts are related, indicated by the weight of the edge, the more one activates the other. Thus, information will spread to closely related concepts easily while distant concepts activate one another only minimally.

Together, the original input and its associations form an association sub-graph which allows us to compare articles based on their conceptual content. The association sub-graph is a directed acyclic sub-graph of the associative network. Rather than copying the edge weights of the associative network, the association sub-graph stores a weight for each node with its activation value. When comparing documents, we use those activation values (Section 3.2) while the graph itself is used for learning (see Section 2.3).

The exact method by which the association sub-graph is constructed can vary. We describe two different methods to construct it from the associative network.

The first method, used in our earlier work [1], is to use spreading activation. In this case, the sum of the weights of all outgoing edges for a node is equal to one. When a node is activated, it spreads the power by which it is activated amongst its outgoing edges according to these weights. Thus, if a node *car* has two outgoing edges, to *racing* with weight 0.75 and to *paint* with weight 0.25, if the *car* node is activated with power 4, it will activate the *racing* node with power 3 and the *paint* node with power 1.

In our research we found that spreading activation sometimes overspreads through nodes that have very few edges, as the activation loses little or no power by spreading through these nodes. A new method which compensates for this is to use a flow network, with some adjustments. A flow network [8] is a directed graph where each edge has a limited capacity and each node receives input from incoming edges called 'flow'. Flow originates from nodes called 'sources' and is absorbed by nodes called 'sinks'. The amount of flow going into a node must equal the amount of flow going out of the node.

An associative flow network acts in a similar manner, with an important difference: in regular flow networks there is a predefined sink, but in an associative flow network every node is a partial sink. This means every node absorbs a small amount of flow, thereby compensating for the problem of overspreading. During an operation, once a node has absorbed this amount, it is saturated, and from then on acts as a regular node in a flow network, passing all incoming flow through the 'pipes'. Based on a certain input of flow, we calculate the flow pattern through the network and take all output nodes that receive flow, ordering them by the amount of flow received. The results are saved along with the flow paths between the input and output nodes.

2.3 Training Method

Each document from a library to be categorized is connected to the associative network by means of the words in the document. When we wish to find the most closely related document, activation is spread from one document (the input document) and documents receive flow or activation as it spreads through the associative network. Documents themselves do not spread flow to other nodes in the network (with the obvious exception of the input document). A human supervisor can then inform the network whether or not the association between those two documents was correct. This feedback is used to train the associative network in a manner very similar to the method used in neural networks: back-propagation.

To implement back-propagation in an associative network, we first take the association sub-graph (see Section 2.2) and reverse all the edges. We then take all nodes that are in the output document and find the set of all paths that lead from those nodes to the source nodes (the observations in the input document). We then prune all edges and vertices that were not part of any of the paths between the output and the observations, as in Figure 4. What remains is a set of all connections between the two documents in the associative network. We reinforce the connections in the trimmed sub-graph if the document pair was correctly linked, increasing their weight in the associative network. Inversely, if the result was incorrect, we weaken those connections, lowering the weight in the associative network. As a result, the network will generate associations along correct lines more quickly, while making associations along incorrect ones less easily.

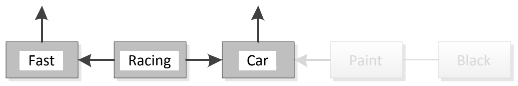

Figure 4. Query flowing back to original input

When a document has been incorrectly linked, we recalculate the association sub-graph after adjusting the network and generate a new result based on the new association sub-graphs, which is again evaluated by the supervisor. This can then be repeated until the association sub-graph produces the correct results. Note that if the correct class is known, no manual action is required.

2.4 Handling Dynamic Data

Many document libraries such as wikis can be easily modified by different users, who can add, remove or edit documents. Any addition, removal or edit can influence the proper way to categorise the documents, which can make maintaining a hierarchical clustering for the library tedious.

The simplest solution to this problem is to rerun the entire algorithm whenever a document is added, removed or edited. Especially for large libraries with frequent changes, such a solution is undesirable as it will require a large amount of resources to keep up with the edit frequency. A better solution lies in dealing with the individual edits separately. If a category becomes too small after removing a document, the hierarchical structure allows the remaining few documents to be grouped in the category above it. The hierarchy guarantees this is valid. Likewise, by calculating the average distance between a new or edited article and the articles in each category, the best match can be found. Finally, when a category becomes too large, it can be split up into multiple subcategories by finding clusters within the data. Associative networks work particularly well with the latter solution because of the independence of the knowledge of the system and the actual connection to existing documents.

All data about proper and improper associations are stored in the network. If a document is removed from the network, no adaptation to the knowledge stored in the associative network needs to be made – this is because the document itself is not stored in the network, merely the relationships between concepts. The assumption here is that even if the document itself is removed, the relationships between concepts extracted from the document are still valid as they are language dependent, not document dependent.

Likewise, when a document is edited, only the relations between the document and the associative network change – the internal structure of the associative network remains the same and thus no knowledge about relationships between words is lost.

An additional feature that helps our method with dynamic data is the fact that the links between two articles are independent of the rest of the library. This allows relations to be categorized in parallel and even allows one part of the library to be reordered without affecting the other parts.

2.5 Performance

One of the advantages of using associative networks is their performance. Regardless of whether flow or spreading activation is used, any vertex in the network can be activated only once. Even if the whole network is activated, the number of activations is thus limited to O(V) where V is the number of vertices. In practice, though it depends largely on the configuration of the flow or spreading activation algorithm, it is far less than that; the associative network is sparse and the nodes activated in a given input are likely to be clustered – in fact, this clustering is why the associative network technique works. This fast performance means the system can respond to changes in the libraries in real time, allowing it to keep a correct categorization at all times.

3 Setting up an associative network

In this section we detail how we used associative networks for our evaluation experiment.

3.1 Creation and Training

As we did in earlier work [1] we used Princeton WordNet [9, 10] to initialise our associative network, using lemmas as nodes. In this earlier work we classified documents in predefined categories, whereas in the current work we create a hierarchical document categorization from scratch. Edges between the lemmas in the associative network were made based on syno-/antonym, hyper-/hyponym, holo-/meronym, troponym and entailment relations and assigned a default weight of 1 as there is no information about the importance of the connection between the lemmas.

WordNet was chosen as it provides a pre-constructed network of terms linked by conceptual meaning. Synsets, sets of synonyms describing the same concept, naturally group together different words expressing the same lemma. The relationships between synsets in WordNet express different types of relations between these lemmas. This combined with the easy availability makes WordNet a good foundation of the associative network.

To link the lemmas expressed by synsets to the words in a document, each node was provided with a automatically generated list of the surface forms (plurals etc.) of each of the synonyms based on the English grammar rules. These surface forms were used in linking the synsets to the actual raw text. No additional NLP techniques were used to improve these links, unlike our earlier work.

Even with surface forms, not all terms in the document are present in Princeton WordNet. Notably proper names and special domain vocabulary are missing and therefore cannot be linked to other terms. One solution would be to provide such a vocabulary for the specific domain covered by the document library and link it up with the WordNet base manually. Alternatively, links could be created based on combined activation patterns of all documents using the term – those terms that receive a high activation often are likely to be linked to the new term. In our case, rather than extend the library or look for links in combined activation patterns, we treated each unknown word as a special instance without relationships to any other word. Thus, these terms were weighted into the final activation pattern based solely on the direct input to that node, without any activation being spread.

Training was done in the same way as our earlier work [1] by creating a training library composed of 30 manually selected Wikipedia articles, with each article being closely related by topic to exactly one article and not related to the other 28 articles. An associative network was initialised using WordNet. After that it was activated for each of the 30 articles in random order to determine which of the articles were the most closely related. Depending on the result, positive or negative reinforcement was applied to the network as described in Section 2.3. This cycle was repeated until the associative network produced the correct matching article for the entire training library.

3.2 Categorization Process

To categorize a library of documents, we started by scanning the text of each document, removing meta-data to acquire the raw text of the document. A list of lemmas corresponding to the words in the document was then generated from the raw text by matching surface forms. If multiple lemmas shared the same surface form, they were all activated. Thus, the word *fast* would activate lemmas for abstaining from food as well as high speeds. The associative network filters out the correct lemma by the activation spread – lemmas that are distant from the rest of the text automatically get less spread from the rest of the input.

The lemmas of each document were then used by the associative network to construct an association sub-graph for each document. Next, these association sub-graphs were compared to determine the distances between documents. This distance was determined based on the total activation value that each node received after association for each document. Specifically, it was calculated as:

$$\sum_{i=1}^{n} |V_{A(i)} - V_{B(i)}| \quad (1)$$

In other words, the sum of the absolute differences between the activation value V of each node 1...n in documents A and B.

Finally, a clustering algorithm based on multi-level graph partitioning [11] was used to identify subsets of documents that were closely related to one another based on the distance between them. Each cluster became a category, together forming a hierarchy. The same clustering algorithm was used for both of the associative networks and the neural network baseline (see Section 4.2).

To determine the name of the category, the association sub-graphs in each cluster were analysed to find the common denominator between the documents, which is the node that received the highest input value after association. As this is the concept that was most shared by all documents in the cluster, it was used as a name for the category.

4 Experiment

To test the effectiveness of our approach, we created two associative networks as described in Section 3.1, one using flow and one using spreading activation. These systems were then used to categorise a library of Wikipedia articles. Finally, those categorizations were compared to a baseline and a gold standard to determine which of them gave the most accurate and most useful results.

4.1 Task

Given libraries of English Wikipedia articles, we wished to find a categorization of the documents in those libraries. No information about the categories to be used was provided beforehand: the system had to create its own categories. However, the system was provided with a guideline for the number of documents a category should roughly contain: around 5 categories for the small libraries and between 300 and 400 categories for the large libraries (see below in Section 4.3). No techniques for balancing between categories were used [12] nor were hierarchies adapted after construction [13]. Categories are hierarchical but a document may only be sorted into one category.

Categorizations were tested automatically and manually. The automatic testing was done by comparing the results to a gold standard, the manual categorization by Wikipedia authors. For the manual evaluation, quality of categorization was rated by human judges using two criteria: usefulness and correctness. First, the categorization has to be useful. Simply declaring a category *universal* and sorting everything in it is accurate but not useful. Likewise, giving each document its own, independent category is not useful. A useful categorization creates sub-categories of roughly similar size, does not subdivide sub-categories that are already very small and does not nest sub-categories too deeply. The composition of the hierarchy should make the information easy to find. Secondly, the categorization has to be correct; this both means that the documents should be assigned to the correct category and that the categories should be grouped correctly. For example a category *flowers* has no business being a sub-category of *vehicles* and a document *swimming techniques of aquatic mammals* does not belong in the category *air-planes*.

4.2 Baseline and Gold Standard

In all of our tests, we compare three different methods of categorization: an associative network based on spreading activation, an associative network based on a flow model, and a neural network baseline. In earlier work we already established that associative networks outperform a TF-IDF baseline [1], so in our current work a comparison to neural networks was chosen due to the structural similarity to associative networks and the shared learning method of backpropagation. Using earlier work on hierarchical document classification [2, 3] as a foundation, we created the baseline by taking a total of twenty large scale neural networks that were constructed and trained analogously with the associative networks, with the network that performed best after training used in the test. Also included in the test is the categorization made manually by the Wikipedia authors, which is used as the gold standard in the automatic evaluation. In Wikipedia, articles can be in more than one category, but for our test we removed all categories other than the one from which the articles were selected (see Section 4.3 below) to leave each article in a single category.

4.3 Libraries

Libraries were generated from a random selection of English Wikipedia articles, with each article representing a document. Articles were selected from different, related subcategories in Wikipedia. The subcategories themselves were selected by first randomly selecting a primary category and then selecting random subcategories recursively to ensure a hierarchical structure. Stub-articles, lists and the likes, and articles with fewer than 1000 words were excluded from the test. Articles (documents) were stripped of all meta-information such as links and categorization and were converted to raw text.

Two types of libraries were generated: small libraries with a small number of articles for evaluation by humans and large libraries with a much larger number of articles, which were only evaluated automatically, due to their size. Sixteen small libraries of articles were constructed, with a total of 290 texts, an average of 18 per library. Sixteen large libraries were also constructed, with between 10.000 and 15.000 articles each. Both sets of libraries were then passed to each of the three algorithms to be categorized.

4.4 Automatic Evaluation

A human-likeness score was generated by an automatically calculated comparison of the resulting categorization with the Wikipedia user base categorization. This score, based on methods for comparing trees [14], was calculated by taking the number of elementary transformations (insert, delete and modify) necessary to morph the result categorization into the Wikipedia user base categorization.

By the human-likeness score, the four methods were ordered from 1 (best) to four (worst). Due to their size, large libraries were evaluated only by means of the human-likeness score; there was no evaluation by human judges.

4.5 Human Evaluation

Since the quality of categorization is often subjective [15], the categorizations of the small libraries were also evaluated by 37 human judges in two ways: on paper (by 12 judges) or via the Internet (by 25 judges). The judges evaluated the categorizations on two criteria: correctness and usefulness. Correctness was defined to the judges as a combination of articles being in the correct category and those categories being correctly named. Usefulness was defined to the judges as the quality of the hierarchy of categories with regards to ordering the articles in groups and the overview the hierarchy provided of the information. The group of judges consisted of men and women from age 20 to 60 with educations ranging from high school level to university educated and backgrounds in Linguistics, Computer Science and Medicine.

Each judge was given the categorizations for a random library and was asked to provide a ranking from one (best) to four (worst) of the three automatic categorizations (neural networks, associative network based on flow and associative network based on spreading activation) as well as the original Wikipedia categorization. An absolute ordering rather than a scoring system (such as each judge assigning a score of 1 to 10 to the two criteria for each categorization) makes the results easier to compare between different judges. The judges were not told which method generated which categorization and the author was not present during the test to ensure double-blindness.

Instructions were given beforehand with a simplified example and judges were asked to review multiple libraries. Each library was reviewed by two judges in paper format and by at least two judges online. No library was reviewed more than once by the same judge. For the off-line evaluation, the results were discussed informally afterwards to get some idea of the reason why certain categorizations were considered better (see Section 5 below).

The method of ranking each categorization was selected based on the subjectivity of the problem and the underlying goal of categorizing documents to allow easier access to new users. These users would generally have only limited knowledge of the topic but would still wish to find information as fast and intuitively as possible. To simulate this in our test, libraries were assigned to judges randomly.

5 Results

In Table 1, the averages of the outcomes of our test are shown. The column labelled *Wiki* is the gold standard categorization made by human Wikipedia authors. The *Flow* and *Spread* columns list the results for the two types of associative networks while the column marked *NN* (neural network) represents the baseline results.

	Wiki	Flow	Spread	NN
Correctness	1.5	2.5	2.9	3.1
Usefulness	1.9	2.1	2.7	3.3
Distance to Wiki	1.0	3.0	2.7	3.4

Table 1. Average rankings over 16 small libraries.

	Wiki	Flow	Spread	NN
Distance to Wiki	1.0	2.9	2.8	3.3

Table 2. Average rankings over 16 large libraries.

The associative networks performed slightly worse than the gold standard of the Wiki categorizations, having a slightly lower correctness – many judges informed the author after their reviews that they found small errors in the categorization that caused them to rate the networks lower on this measure. In several cases the subdivision was considered good, but the name assigned to the category did not reflect the content.

In regards to usefulness, associative networks based on flow were especially successful, getting close to the gold standard Wiki categorization. The general goal of the categorization as mentioned in the introduction is the ordering of the information, but various judges stated a different task, such as searching for the answer to a specific question, might have influenced their decision.

Comparing the results of small libraries in Table 1 to those of large ones in Table 2, we can see that the distance to the human categorization is fairly similar. Based on this we expect that if the large libraries were evaluated by humans, the results on correctness and usefulness would also be similar to those of the small libraries.

6 Related Work

Text categorization has been approached from many different angles. TF-IDF [16] and other Vector Space models [17] use similarity in words to categorize documents using techniques such as Support Vector Machines [18]. These primarily consider the presence or absence of keywords and make a statistical analysis of word frequencies. As such, they are unable to draw upon the conceptual meaning of the text, which limits their ability to find matches. For example, the sentences *the fast black car was winning* and *the speedy dark vehicle was victorious* have roughly the same meaning, but do not share any important words. Associative networks are not limited in this way; they will produce a nearly identical association sub-graph for the two above sentences.

Latent Semantic Analysis [19] is a vector based approach that appears at first glance to share some properties with associative networks, notably in that they both attempt to extract the deeper meaning from the text through linking related words. While associative networks receive these links as input and find weights for them, LSA constructs the links from training data. This can lead to good results, but LSA may easily make invalid connections between words that happen to coincide in the training data by chance. Associative networks, relying on an existing, quality network where relationships are known to be valid, does not suffer from such problems. Additionally, LSA is unable to deal with negation [19] – a natural result of the bag-of-words approach. Like LSA, associative networks use a bag-of-words, but since WordNet links antonyms together, activation spreads over the negation barrier easily. For example, in the sentence *the Ferarri was not a slow car* the word *fast* will receive activation not just from the words *Ferarri* but from the word *slow* as well. Even with a different basis than WordNet, links such as this can be added to the associative network easily.

Ferilli et al. [20] propose two methods which rely on word co-occurrence that share some similarities with associative networks, but like LSA, they rely on finding valid connections through co-occurrence, which makes it subject to the same problem of potentially invalid connections through coincidental co-occurrence. It should additionally be noted that associative networks operate in O(v) (see Section 2.5) worst case and a fraction of this on average due to the sparsity of the network and are computationally more efficient than the performance listed for both LSA and Ferilli's methods. This makes associative networks far more viable for a live environment with large libraries of documents being edited, added and deleted by multiple users simultaneously.

Other solutions for text categorization are knowledge-based systems that use pre-existing domain knowledge such as decision trees [21]. Such systems require additional data about the problem space, which must be provided for each domain, while an associative network will work on any set of documents and the only information that is required is knowledge of the language in which the documents are written to initialize the network.

Like associative networks, concept mining [22] is able to use the underlying meaning of the text to find relationships between documents. This and other approaches based on NLP [23] have produced good results, but these require more information than associative networks, for example regarding grammar and sentence structure. This makes these systems more difficult to construct and some have limitations when dealing with short sentences or incorrect grammar. By contrast, associative networks do not require any knowledge of grammar and they do not need to correctly parse sentences to determine their deeper structure. Rather associative networks require only the words in the text which can be determined swiftly and easily.

There are some hybrid systems [24] that combine several of these techniques – associative networks could be incorporated in such systems as well. Finally, various systems resembling associative networks exist, some also founded on psychological models [25], while others use similar knowledge of the underlying concepts in the text to aid in categorization [26, 27]. The latter stay closer to the actual text than associative networks do, however.

7 Conclusion and Future Work

In our evaluation experiments, the associative networks were found to perform consistently better than the baseline neural network, with the new flow network based associative networks being a possible improvement on spreading activation based associative networks and at the very least producing similar quality results. Furthermore, human judges rated the categorizations created through associative networks close to the gold standard in terms of usefulness and correctness.

Another advantage of associative networks is their performance – the quality combined with the speed of associative networks and their ability to deal with dynamic data makes them an excellent solution for automatically categorizing large dynamic libraries even in a live environment where documents may be edited constantly. In future work, we want to expand on the handling of dynamic data, specifically testing the usability in such a live environment.

Though in earlier work we compared the performance of associative networks to TF-IDF, no comparison like the one for neural networks has been made to any of the methods for categorization mentioned in Section 6, so doing so will be the next step in our research. Correctly naming the categories using the associative network is another topic that merits further research; see e.g., [30].

We believe our agglomerative clustering algorithm [11] to be particularly well suited as it groups closely related documents together very easily which is important for a high correctness. Alternative clustering algorithms, such as the divisive method of [28] may allow more diversely shaped clusters of documents to be found more easily, while different methods of merging or splitting clusters during the construction of the hierarchy can significantly affect the final hierarchy [29]. As our focus was on the associative network, not the effect of different clustering algorithms, examining this effect is left as future work.

References

[1] Bloom, N., Using natural language processing to improve document categorization with associative networks *Proceedings of the 17th international conference on Applications of Natural Language Processing and Information Systems*, Groningen, The Netherlands, pp. 117-182, 2012 Own work, suppressed for review

[2] Jeschke, G.,Lalmas, M., Hierarchical Text Categorisation based on Neural Networks and Dempster-Shafer Theory of Evidence *EUROFUSE Workshop on Information Systems*, Villa Monastero, Italy, pp. 23-25, 2002

[3] Lee, H.M., Chen, C.M., Hwang, C.W., A Hierarchical Neural Network Document Classifier with Linguistic Feature Selection *Applied Intelligence*, Vol 23, pp. 277-294, 2005

[4] Marcus, G. F., *The Algebraic Mind: Integrating Connectionism and Cognitive Science* Cambridge, MA: MIT Press., ISBN: 0-262-13379-2, 2001

[5] Schank R. C., *Dynamic Memory: A Theory of Learning in Computers and People* New York: Cambridge University Press, ISBN: 0-521-24858-2, 1982

[6] Schank, R.C., Abelson, R.P., *Scripts, Plans, Goals and Understanding* Erlbaum, Hillsdale, New Jersey, U.S., ISBN: 0-470-99033-3, 1977

[7] Bechtel, W., Connectionism and the philosophy of mind: an overview *The Southern Journal of Philosophy*, Vol 26, pp. 1741, 1988

[8] Ford, L. R., Fulkerson, D. R., *Flows in Networks* Princeton, NJ: Princeton University Press, 1962.

[9] Miller, G., WordNet: A Lexical Database for English *Communications of the ACM*, Vol 38, No 11, pp. 39-41, 1995

[10] Fellbaum, C., *WordNet: An Electronic Lexical Database* Cambridge, MA: MIT Press, 1998

[11] Karypis, G., Kumar, V., A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs *SIAM Jour. on Scientific Computing*, Vol 20, pp. 359-392, 1998

[12] D'Alessio, S., Murray, K., Schiaffino, R., Category Levels in Hierarchical Text Categorization *Proc. of the Third Conference on Empirical Methods in Natural Language Processing (EMNLP-3)*, 1998

[13] Li, T., Zhu, S., Hierarchical Document Classification Using Automatically Generated Hierarchy *Proc. of the SIAM International Conference on Data Mining*, Newport Beach, CA, USA, 2005

[14] Zhang, K., Shasha, D., Simple fast algorithms for the editing distance between trees and related problems *Siam Jour. of Computing*, Vol 18-6, pp. 1245-1262, 1989

[15] Sebastiani, F., Text Categorization *Text Mining and Its Applications*, pp. 109-129, WIT Press, 2005.

[16] Ramos, J., Using TF-IDF to Determine Word Relevance in Document Queries *International Conference on Machine Learning*, 2003

[17] Soucy, P., Mineau, G.W., Beyond TFIDF Weighting for Text Categorization in the Vector Space Model *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, 2005

[18] Cai, L., Hofmann, T., Hierarchical Document Categorization with Support Vector machines *Proc. of the Conf. on Information and Knowledge Management*, 2004

[19] Wiemer-Hastings, P., Latent Semantic Analysis *Proceedings of the 16th international joint conference on artificial intelligence*, Vol 2, pp. 932-937, Morgan Kaufmann Publishers Inc. San Francisco, CA, USA 1999

[20] Ferilli, S., Biba, M., Basile, T.M.A., Esposito, F., Using Explicit Word Co-occurences to Improve Term-Based Text Retrieval *Com. in Computer and Inf. Science*, Vol 91, pp. 125-136, Springer Heidelberg, 2010

[21] Apte, C., Damerau, F., Weiss, S.M., Text Mining with Decision Trees and Decision Rules *Conference on Automated Learning and Discovery*, Carnegie-Mellon University, June 1998

[22] Shehata, S., Concept Mining: A Conceptual Understanding based Approach *PhD Thesis in Electrical and Computer Engineering*, University of Waterloo, Ontario, Canada, 2009

[23] Ekedahl, J., *Danish Natural Language Processing in Automated Categorization* Lunds universitet, LTH, MSc. Thesis, March 2008.

[24] Forman, G.H., Suermondt, H.J., Hierarchical categorization method and system with automatic local selection of classifiers *U.S. Patent 7349917* March 25, 2008

[25] Wichert, A., Hierarchical Categorization *Proceedings of the Midwest Artificial Intelligence and Cognitive Science (MAICS)*, 1998

[26] Bang, S.L., Yang, J.D., Yang, H.J., Hierarchical document categorization with k-NN and concept-based thesauri *Information Processing and Management* Vol 42, issue 2, pp. 387-406, 2006

[27] Tikk, D., Yang, J.D., Bang, S.L., Hierarchical Text Categorization using fuzzy relational thesaurus *Kybernetika*, vol. 39, issue 5, pp. 583-600, 2003

[28] Zhong, C., Miao, D., Franti, P., Minimum spanning tree based split-and-merge: A hierarchical clustering method *Information Sciences*, Vol 181, No 16, pp. 3397-3410, 2011

[29] Ding, C., He, X., Cluster merging and splitting in hierarchical clustering algorithms *IEEE International Conference on Data Mining (ICDM'02)*, Japan, 2002

[30] Fukumoto, F. and Suzuki, Y., Cluster Labeling based on Concepts in a Machine-Readable Dictionary *Proceedings of 5th International Joint Conference on Natural Language Processing*, pp. 1371-1375, November 2011