International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:5, No:11, 2011

Using Suffix Tree Document Representation in Hierarchical Agglomerative Clustering

Daniel I. Morariu, Radu G. Creţulescu, Lucian N. Vinţan

***Abstract*—**In text categorization problem the most used method for documents representation is based on words frequency vectors called VSM (Vector Space Model). This representation is based only on words from documents and in this case loses any "word context" information found in the document. In this article we make a comparison between the classical method of document representation and a method called Suffix Tree Document Model (STDM) that is based on representing documents in the Suffix Tree format. For the STDM model we proposed a new approach for documents representation and a new formula for computing the similarity between two documents. Thus we propose to build the suffix tree only for any two documents at a time. This approach is faster, it has lower memory consumption and use entire document representation without using methods for disposing nodes. Also for this method is proposed a formula for computing the similarity between documents, which improves substantially the clustering quality. This representation method was validated using HAC - Hierarchical Agglomerative Clustering. In this context we experiment also the stemming influence in the document preprocessing step and highlight the difference between similarity or dissimilarity measures to find "closer" documents.

***Keywords*—**Text Clustering, Suffix tree document representation, Hierarchical Agglomerative Clustering

I. Introduction

WHILE more and more textual information is available online, effective retrieval is difficult without good indexing and summarization of document content. Document categorization is one solution to this problem. The task of document categorization is to assign a user defined categorical label to a given document. In recent years a growing number of categorization methods and machine learning techniques have been developed and applied in different contexts.

Documents are typically represented as vectors in a features space. Each word in the vocabulary is represented as a separate dimension. The number of a certain word's occurrences in a document represents the value of the corresponding component in the document's vector [1, 3].

In this paper we present a comparative study for improving clustering results by representing documents using the suffix tree document model (STDM). This representation takes into account the order of words in the sentence rather than, their frequency.

The suffix tree document model appears in the literature related to the Suffix Tree Clustering (STC) algorithm [3, 8]. In this algorithm the documents are represented as suffixes of phrases in the suffix tree. In the suffix tree the nodes represent the common words of the documents and the leaves are all the words from documents. The researches usually are focused to improve the STC algorithm itself [8]. In the literature the most investigated in the STC is the score of suffix tree nodes which become the clusters.

In this paper we propose that in the step of document representation to use the Suffix Tree Document Model (STDM) and then use the data represented in this manner in a clustering algorithm. Our idea is to use an algorithm that is based on distance matrixes. We have decided to use a hierarchical clustering algorithm. We decided this because we do not want to build the STDM model tree for the entire document collection; instead we will build one suffix tree for any two documents from the data set and compute the similarity between them. The advantage of this method is that the resulting tree size is much smaller than the tree for the entire data set.

We propose to examine here whether using the STDM model to represent documents in clustering algorithms can improve the clustering's results, compared with using the classical vector representation model (VSM - Vector Space Model). The STDM representation regarding the representation of text documents includes besides the words from the document, also the order of words from the phrase. Thus, by default, this model contains some elements of syntax of the documents. Although the STDM is a computational representation, by representing words order in a tree, brings us somehow to a closer "ontological" documents representation. This was the scientific hypothesis on which we started this research. In the STDM model we will represent the entire documents with all the phrases and all the words from the phrases. The nodes in the tree will contain common words or phrases from both documents. If the tree has more common nodes then we obtain a greater similarity between the two documents. Additionally, we propose in our formula to take into account the number of common words in each node, too.

Section 2 and 3 contains prerequisites for the main work developed in this approach, presenting the document representation model and clustering algorithms used. In section 4 we present the methodology used for computing the similarity matrix and our proposed formula. Section 5 presents

Daniel I. Morariu is with "Lucian Blaga" University of Sibiu, Engineering Faculty,Computer Science Department, E. Cioran Street, No. 4, 550025 Sibiu, ROMANIA, (email: daniel.morariu@ulbsibiu.ro).

Radu G. Cretulescu is with "Lucian Blaga" University of Sibiu, Engineering Faculty,Computer Science Department, E. Cioran Street, No. 4, 550025 Sibiu, ROMANIA, (email: radu.kretzulescu@ulbsibiu.ro).

Lucian N. Vintan is with "Lucian Blaga" University of Sibiu, Engineering Faculty,Computer Science Department, E. Cioran Street, No. 4, 550025 Sibiu, ROMANIA, (email: lucian.vintan@ulbsibiu.ro).

the experimental framework and section 6 presents the main results of our experiments. Finally the last section debates and concludes on the most important results obtained and proposes some further work.

II. Document Representation

A. Vector Space Model (VSM)

In the Vector Space Model (VSM) a text document is represented as a vector of terms with associated frequencies [7, 9]. The term definition is not imposed by the model, but the terms are usually words or word sequences. If words are chosen as terms, from the entire dataset it is created a dictionary with all words from the documents (the vocabulary V) where each word is an independent dimension of a k-dimensional vocabulary. Any text document can be represented by a high dimensional vector in this space.

Whether $d_i = (w_{i1}, w_{i2}, ..., w_{ik})$ the representation of a document where w_{ij} is the weight of term j (from the vocabulary V) in document d_i.

There are several ways of representing the weights of attributes from the vector [3]. Depending on the chosen representation, some learning algorithms work better or worse. In the experiments we have used *Nominal representation* where in the input vector we compute the value of the weight using the formula:

$$TF(d,t) = \frac{n(d,t)}{\max_\tau n(d,\tau)} \quad (1)$$

where *n(d, t)* is the number of times that term *t* occurs in document *d*, and the denominator represents the value of term that occurs the most in document *d*, and *TF(d,t)* is the term frequency.

B. Suffix Tree Data Model - STDM

In the Suffix Tree Data Model (STDM) model documents are represented as suffix trees in which the nodes are documents with common words and the leaves are all the words from the documents [3,4].

Let $d=c_1c_2c_3...c_m$ be a representation of a document as a sequence of words c_i with $i = 1, m$ and a set S of n documents. The suffix tree for a set S containing n phrases, each of m_n length is a tree that contains a root node and exactly $\sum m_n$ leaves.

Rules for building the suffix tree are:

- Each internal node other than root must have at least two children and each edge leaving a node is labeled with a nonempty substring of n.
- Any two edges that start from the same node cannot start with the same word.

Since each node contains at least two children it will contain the common part for at least two suffixes. All branches from the root node to the leaf are one document or a phrase in a document. As two documents have more and more common nodes these documents tend to be more similar.

III. Clustering algorithm

In this paper we have decided to use the hierarchical algorithm HAC - Agglomerative Clustering Hierarchical. Thus in our STDM representation it is not needed to build the tree for the entire collection of documents, such as the current approaches [3, 4]; instead of doing this, we will propose to build one suffix tree for any two documents from the data set. This method has the advantage that the resulting tree size is much smaller than the tree for the entire data set. However this method has the disadvantage to build $n(n-1)/2$ small trees, but the construction time required for such a smaller tree is considerably reduced because the trees have few branches and then the search is much faster. Also we need to build and search only in one tree at a time and therefore the required memory is much smaller, too. All approaches that build suffix tree for all documents (see algorithm STC) finally use methods of disposing nodes so that the search can be made in a reasonable time. Removal of nodes can lead to loss of useful information. Because our trees are small, we will not use any nodes removing methods.

For the clustering we use an algorithm based on the distance matrix. This matrix is calculated initially and, after starting the clustering algorithm this matrix does not change substantially. We chose this approach because in STDM representation it is difficult to represent (compute) a "medoid" (sometimes is the center of gravity of some documents) and restore the distance matrix.

A. Hierarchic Agglomerative Clustering (HAC) – single link type

In the HAC algorithm, at the beginning, each document is considered as forming a cluster and then in the next level (a more general level) two clusters are joined based on their similarity. The algorithm is repeated until all clusters are joined into a single cluster. The dissimilarity between clusters is expressed as the distance between two clusters. For example, the distance can be computed based on Euclidean distance or cosine angle between two vectors.

In the single link approach the similarity between two clusters is given by the minimum distance between the most similar two documents contained in those clusters. For example for two clusters A and B with document $a \in A$ and document $b \in B$ then:

$$sim(A,B) = \min_{a \in A, b \in B} sim(a,b) \quad (2)$$

For the HAC algorithm we use the AGNES (AGglomerative NESting) implementation that starts with the positioning of each document in its own cluster - thus at beginning the number of clusters is equal to the number of documents - then two clusters are joined at a time until the stop criteria is accomplished or until we get a single cluster containing all documents.

The AGNES algorithm is a bottom-up approach in terms of building clusters. The pseudo code for the AGNES algorithm is:

AGNES ALGORITHM

Input: *n* objects	
Output: *k* clusters	
Step 1:	Note all documents from 1 to *n*. Each document is a cluster;
Step 2:	Compute the distance *d(r, s)* between objects *r* and *s* with *r, s = 1, 2,..., n*; be D = (*d(r, s)*) the similarity matrix
Step 3:	Find most similar clusters *r, s,* so *d(r, s)* is minim in D;
Step 4:	Join *r* and *s* into a single new cluster note *t*. Compute *d(t, k)* for all clusters $k \neq r,s$. Delete the row and column for clusters *r* and *s* from *D* and add a new column and row for the new cluster *t;*
Step 5:	Repeat from step 3 until the stopping criterion is accomplished.

The structure returned by the algorithm provides more information than a set of clusters returned by the commonly algorithms. AGNES clustering result is a hierarchical structure in which the more we are on a higher level we obtain the bigger and more general clusters. The result of the algorithm is a dendrogram. The AGNES clustering algorithm does not require specifying, at the start, the number of clusters that are needed to be obtained. The disadvantage of this type of clustering is that it is less effective given the complexity at least quadratic on the number of documents.

IV. THE SIMILARITY MEASURE

Both algorithms require the computation of the *distance matrix* that contains the distances between any two documents. To calculate distances between any two distinct documents from the dataset we have chosen two types of measures:

- Measures that describe the dissimilarity between documents - in this category we implement two distinct measures - called Euclidean distance and Canberra Distance;
- Measures that describe the similarity between documents – in this category we implement three type of measures: called Jaccard Distance, OLDST Distance and our proposed distance called NEWST Distance;

For the first three distances (Euclidean, Canberra and Jaccard) we use the VSM representation of documents and for the last two distances we use STDM representation of documents. The first distance category that represents dissimilarity returns results in $[0,\infty)$ domain and we convert them into [0,1] domain.

A. Euclidean Distance

$$d_{Euc}(x_i, x_j) = \sqrt{\sum_{k=1}^{p} (x_{ik} - x_{jk})^2} \quad (3)$$

B. Canberra Distance

$$d_{CAN}(x_i, x_j) = \sum_{k=1}^{p} \frac{|x_{ik} - x_{jk}|}{|x_{ik}| + |x_{jk}|} \quad (4)$$

where x_i, x_j are documents from the dataset and p is the number of elements (words) from vocabulary that represent the documents.

C. Jaccard Distance

$$d_{Jac}(x_i, x_j) = \left(1 - \frac{\#common\_elements}{\#total\_elements}\right) \quad (5)$$

where x_i, x_j are documents from the dataset, *#common_elements* – represent the number of elements (words) that occur in both documents using VSM representation, and *#total_elements* – represents the total number of elements needed for representing the documents x_i and x_j – the total number of distinct elements from both documents.

For this metric the domain is [0,1], and because it represents the similarity between documents we decrease the result from 1 so it can be used in the dissimilarity matrix. Also for the following two distances we have converted the similarity to dissimilarity decreasing the result from 1.

D. OLDST Distance

This distance use the STDM representation of documents and is a most used method for computing similarity in this type of representation. In the document representation using STDM model we build separate one suffix tree for any two documents for the entire data set and we will calculate the distance between them using one of the formulas OLDST and NEWST.

$$d_{OLDST}(x_i, x_j) = \left(1 - \frac{\#common\_nodes}{\#total\_nodes}\right) \quad (6)$$

This distance returns the results in the [0,1] domain with 0 most similar.

E. NEWST Distance

The NEWST distance is proposed by us and it is used with the STDM representation of documents.

$$d_{NEWST}(x_i, x_j) = \left(1 - \frac{1+\#common\_nodes}{1+\#total\_nodes}\right) * \frac{1}{1+\#words\_from\_node} \quad (7)$$

where *#common_nodes* represents the number of nodes from the suffix tree that are contained in both documents; *#total_nodes* represents the total number of nodes from the suffix tree; *#word_from_node* represents the total number of elements (words) that are traversed to reach a common node;

For this type of representation in case when we have no common nodes the results tends to 1, reach 1 when the total number of nodes tends to infinity. And the result is 0 when all nodes in the tree are common to both documents.

Our proposed formula for computing the NEWST distance has the following advantages. First: if two documents have no common nodes then the distance between them depends on the number of words that are used to represent those two documents. If the documents are larger and do not even have common nodes, the distance between them will be closer to 1 but still different depending of the documents dimension. Compared with other metrics that always return the value 1 if the documents have no common nodes, this small difference helps us to determine the order in which documents will be joined based on the distance matrix. Such the large documents will be merged last. The second advantage: if the documents have common nodes we have weighted the returned value with

the number of words that are in the common node. This we can make the difference between documents that have small common parts and documents that have large common parts.

This two advantages offer us the possibility to make a more accurate clustering.

V. Experimental Framework

A. The Dataset

For evaluating the algorithms we have created data sets containing news obtained from Reuters and the BBC Agency's website [5, 6]. The news were selected from several RSS feeds (only XML files), from 3 consecutive days, and have been pre-classified by the agencies. We have selected the following news category: *Libya, Motorsport, Japan, Israel, Syria, Yemen* and *EuropeanFootball*. In the preprocessing step from xml files we have extracted only the <*title*> node that contains the title of the news and <description> node that contains the body of the news and we have saved separately each news.

In the first phase of preprocessing we have removed special characters that appear in the news for reasons of Web interpreter and finally, in text files, on the first line we save the title and the remained lines contain the news - sentences separated by dots. We kept the category already established by news agencies. We will use this information in the final stage for evaluation the clustering results.

From all of the 193 documents obtained initially grouped into seven categories, we created seven training sets that differ in the number of categories and the number of documents.

After the preprocessing step, we have generated two distinct datasets: one dataset having seven sets where we extract de root words in the preprocessing step and one dataset having seven sets where we don't extract de root of the words in the preprocessing step. For the root extraction we have used the Porter implementation algorithm.

B. Evaluation of the cluster's results

In the cluster evaluation process we use, as base classes, the category from which the news was extracted. Thus we can use for evaluating the results of the clustering algorithms measures like accuracy and f-measure score. Those two measures are generally used in the evaluation of clustering and classification algorithms for pre-labeled data [2]. Because in the step of news selection we have chosen only pre-labeled data and the documents from the data sets were saved with the name of the category, we considered to have a pre-labeled datasets. We believe that the labeling done by the news agencies sites is "perfect" and we will relate to it. The name of the class is given by the label that appears most frequently in that cluster. If there is a cluster already labeled with that value then we will use the next valid label or we will specify "No class" if we do not have available labels.

The *Accuracy* represents the percentage of documents that are correctly grouped in categories based on document labels. *F-measure* is a measure that combines precision and recall and is computed as:

$$F\_measure(C_i, S_j) = \frac{2 * precision(C_i, S_i) * recall(C_i, S_i)}{precision(C_i, S_i) + recall(C_i, S_i)} \quad (8)$$

where C_i is a cluster and S_j is a cluster label. We prefer F-measure score because it is a weighted average that includes and balances the influence of two measures (precision and recall).

Precision – is the percentage of retrieved documents that are in fact relevant to a query.

$$precision(C_i, S_j) = \frac{|C_i \cap S_j|}{|C_i|} \quad (9)$$

Recall – is the percentage of documents that are relevant to the query and were in fact retrieved.

$$recall(C_i, S_j) = \frac{|C_i \cap S_j|}{|S_i|} \quad (10)$$

In fact *precision* represents a quantitative measure of the information retrieval system while *recall* represents a qualitative measure of this system. At the final, the general measure F-measure for evaluating a clustering solution is weighted by the relative size of each cluster.

$$F\_measure(S) = \sum_i \frac{n_i}{n} \max\left(F\_measure\left(C_i, S_j\right)\right) \quad (11)$$

where n_i is the number of documents in cluster *i* and *n* is the total number of documents in the dataset. The value of *F_measure(S)* belongs to the [0,1] interval; the more the value approaches to 1 means a better result.

While accuracy for current category takes into account only the number of documents correctly grouped in that category, F-measure takes into account in addition the number of documents grouped in categories other than the current category and really not need to be grouped in the current category. Thus F-measure is a finer measure of quality data grouping.

VI. Experimental Results

All results that are presented below were obtained using a computer with Intel (R) Core2 Duo T6600 processor at 2.20 GHz, 4GB DRAM and Windows 7 Home Premium operating system.

We will present detailed results on RSS datasets obtained by the HAC (Clustering Agglomerative Hierarchical). The results are presented separately for each data representation model (VSM-Vector Space Model and STDM-Suffix Tree Document Model). Finally we present some comparative results between the two types of representations. Also, for each model of representation we will present results on the dataset without/with applying stemming. For evaluation of the clustering quality we will use accuracy and F-measure score, measures that were presented in Section V.B. We will use both measures to ensure the validity of the clustering results. Accuracy and F-measure score should reflect basically the same trends. Results for accuracy are presented as percentages and the results for F-measure score will be numbers in the interval [0, 1] value of 1 means the best value.

In the Table 1 we present all obtained results, from accuracy point of view for each dataset separately and in last columns

TABLE I
RESULTS OBTAINED BY HAC ALGORITHM - ACCURACY

Data Repr-esentation	Stemming	Metric	Data Sets S01	S02	S03	S04	S05	S06	S07	Average
VSM	No	Jaccard	71.52%	53.80%	66.67%	79.10%	31.09%	65.52%	79.33%	63.86%
	Yes	Jaccard	41.72%	33.92%	39.10%	34.46%	33.68%	68.97%	34.64%	40.93%
	No	Euclidian	41.06%	35.09%	39.74%	34.46%	32.64%	36.21%	34.64%	36.26%
	Yes	Euclidian	43.05%	35.09%	39.74%	34.46%	32.64%	39.66%	34.64%	37.04%
	No	Canberra	42.38%	34.50%	39.74%	33.33%	32.12%	25.86%	32.96%	34.42%
	Yes	Canberra	42.38%	33.92%	39.10%	33.33%	32.12%	25.86%	32.96%	34.24%
STDM	No	OLDST	**100.00%**	**100.00%**	65.38%	**96.05%**	72.02%	70.69%	**96.09%**	85.75%
	Yes	OLDST	**100.00%**	**100.00%**	65.38%	**96.05%**	72.02%	70.69%	**96.09%**	85.75%
	No	NEWST	**100.00%**	99.42%	**95.51%**	77.40%	**76.17%**	**84.48%**	77.65%	**87.23%**
	Yes	NEWST	**100.00%**	99.42%	**95.51%**	77.40%	**76.17%**	**84.48%**	77.65%	**87.23%**

we present the average obtained over all datasets. Also it presents results for the data sets with and without applying stemming in the preprocess step and using all metrics for distance, each metric with type of appropriate representation.

With bold we mark the better results obtained for each set. As it can be observed the STDM representation obtains better results all the time. From the results shown in Table 1 we noted that using the STDM model the accuracy of clustering algorithm is not affected by the extraction or not of words root in the preprocessing step. This is understandable because the extraction of roots didn't lost the word order that somewhat will keep the syntax of document. For the VSM model words root extraction is beneficial only for Euclidean distance and there is a small improving, with only 0.78% in accuracy. For other distances used in the VSM model there is a worsening of 0.18% in Canberra distance and 22.93% in the Jaccard distance. The last one is understandable because the Jaccard distance is based on the number of common elements between the two documents and with roots extraction we obtain more identical words, thereby losing, in this case, the difference between documents and so that the accuracy suffered.

The results presented above show that the STDM model used to represent text documents is applicable to other clustering algorithms, than STC. Also an advantage brought by our implementation is that the suffix tree is calculated only for two documents that reduces time and memory requirements.

Introducing some syntactic elements in document representation - in this case the words order - led to a significant improvement in clustering. We show in Fig. 1 averages results obtained over all seven data sets as accuracy and in Fig. 2 the averages obtained from F-measure point of view for each used metrics. As it can be observed in fig. 2 we obtain the best results again for STDM representation; also the NEWST distance obtain better result for almost all datasets. In case with stemming we obtain with OLDST metric the results with 0.016 better, and form NEWST metric the difference is unnoticed.

From the results presented in the figures above we can observe that in case of using STDM model the quality of clustering have average improvements of 46.30% in accuracy and 0.4121 for F-measure score.

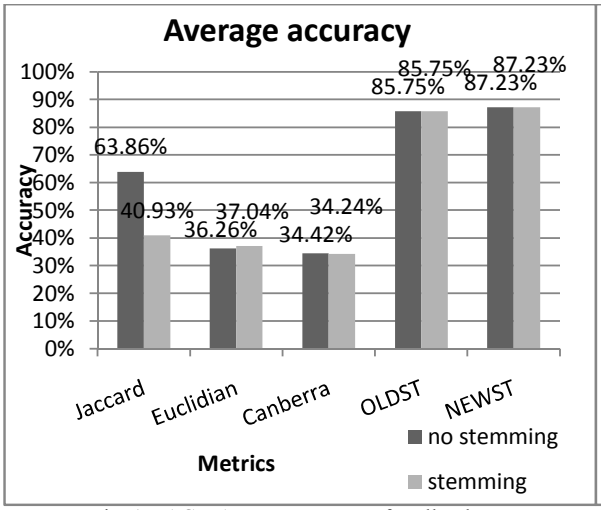

Fig. 1 HAC – Average accuracy for all 7 datasets

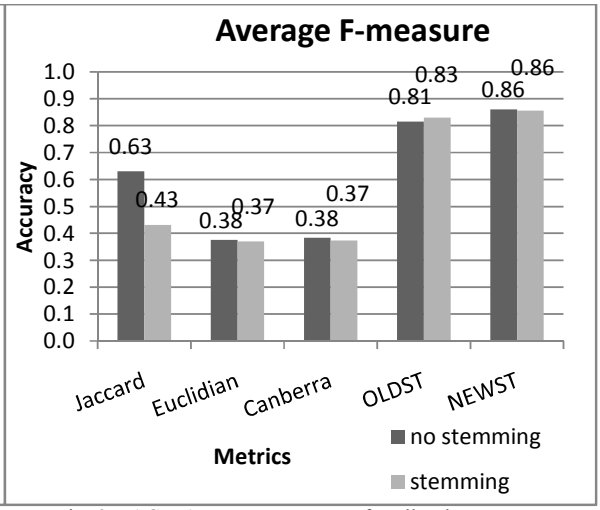

Fig. 2 HAC – Average F-measure for all 7 datasets

VII. Conclusions and Further Work

In this paper we have examined the possibility of using in the clustering algorithm like HAC the STDM model (Suffix Tree document model) for representation of documents and we compared the results with results obtained when we used the VSM (Vector Space Model) representation of documents. For suffix tree model we have developed two formulas for calculating the similarity between two documents, called OLDST and NEWST. For the VSM model we have used for computing the distance between two documents the Euclidean distance and Canberra distance (which are measures of dissimilarity), and Jaccard measure that is a similarity measure.

For evaluation of the results we have used two metrics: the accuracy and the F-measure score. The dataset used comes from specific feed news from Reuters and BBC agencies and each news item is pre-labeled and we consider that is "perfect".

The Suffix Tree representation model for the documents proved to be superior to the vector model noticing an improvement on average for 34.84% of HAC algorithm, the maximum recorded is for the S02 dataset where there is an improvement of 55.56%. Also to this result contributed the OLDST and NEWST formulas of similarity used by us. Another interesting observation is that the similarity measures obtain better results comparing with dissimilarity measures.

We propose a new metric (called NEWST) used to calculate the similarity between two documents represented in the STDM model. This metric applied to STDM model representation for HAC clustering algorithm obtained on data sets S01-S07 an average improvement of 34.84% from accuracy of clustering point of view compared to Jaccard metric used in VSM representation. Average accuracy on S01-S07 sets calculated for metric NEWST with STDM model representation was of 87.23% comparatively to 52.39% that was obtained with Jaccard metric and VSM representation.

As further work it would be interesting to use clustering algorithms from other categories like partitional category, algorithms that didn't allow building overlapped clusters. Also it is interesting to change the representation of STDM for clustering algorithms in order to contain more powerful semantic information. Developing some domain ontologies to guide the clustering algorithm would be very useful for improving results.

Acknowledgment

This work was partially supported by CNCSIS-UEFISCSU, project number PN II-RU code PD_670/2010.

References

[1] S. Chakrabarti, Mining the Web- Discovering Knowledge from hypertext data, Morgan Kaufmann Press, 2003.
[2] Kaufman, L. and Rousseeuw, P.J. Finding Groups in Data: An Introduction to Cluster Analysis, Wiley-Interscience, New York (Series in Applied Probability and Statistics), 1990
[3] Manning, C., Raghavan, P., Schütze, H. Introduction to Information Retrieval, Cambridge University Press, ISBN 978-0-521-86571, 2008
[4] Meyer,S., Stein, B., Potthast, M., The Suffix Tree Document Model Revisited, Proceedings of the I-KNOW 05, 5th International Conference on Knowlegdge Management, Journal of Universal Computer Science, pp.596-603, Graz, 2005
[5] http://feeds.bbci.co.uk/news/rss.xml
[6] http://www.reuters.com/tools/rss
[7] Salton, G., Wong, A., Yang, C. S., A vector space model for information retrieval. Communications of the ACM, 18(11), 1975.
[8] Janruang, J. Guha, S., Semantic Suffix Tree Clustering, In Proceedings of 2011 International Conference on Data Engineering and Internet Technology (DEIT 2011), Bali, Indonesia, 2011.
[9] Morariu, D., Text Mining Methods based on Support Vector Machine, MatrixRom, Bucharest, 2008.

Daniel I. Morariu, PhD, was born at September 17th 1974 in Sighisoara, Romania. He graduates "Lucian Blaga" University of Sibiu, obtaining a M.Sc. in Computer Engineering, and a Ph.D. in Computer Science from the same university. The PhD title is "Contributions to Automatic Knowledge Extraction from Unstructured Data", PhD supervisor Professor Lucian N. VINTAN. The PhD program was partially supported from scientific and financial point of view by SIEMENS Corporate Technology from Munich. At present he is a full-time lecture at "Lucian Blaga" University of Sibiu, Engineering Faculty, Computer Science department. He published over 14 scientific papers in international conference from Romania, Czech Republic, Nederland and Spain.

Radu G. Creţulescu, was born at August 8th 1968 in Sibiu, Romania. He graduates "Babes-Bolyai" University of Cluj-Napoca and obtained a M.Sc. in Computer Engineering, from the "Lucian Blaga" University of Sibiu. At the present time he is a full-time lecture at University of Sibiu, Engineering Faculty, Computer Science department. He published over 8 scientific papers in international conferences from Romania, Finland and Greece.

Profesor Lucian N. Vinţan, PhD ("Lucian Blaga" University of Sibiu, RO) is an active researcher in Advanced Computer Architecture, Context Prediction in Ubiquitous Computing Systems, Text Documents Classification, etc. He is a member of Academy of Technical Sciences from Romania, European Commission Expert in Computer Science, and Visiting Researcher Fellow at University of Hertfordshire, UK. Professor Vintan published 8 books and over 100 scientific papers (Romania, USA, Italy, UK, Portugal, Hungary, Austria, Germany, Poland, China etc.). For his merits, Professor Vintan obtained the "Tudor Tanasescu" Romanian Academy Award. He introduced some well-known original architectural concepts in Computer Architecture domain (Dynamic Neural Branch Prediction, Pre-Computed Branches, Value Prediction focused on CPU's Context, etc.), recognized, cited and debated through over 80 papers published in many prestigious international conferences and scientific reviews (ACM, IEEE, IEE, etc.)