EU-IST Project IST-2003-506826 SEKT
SEKT: Semantically Enabled Knowledge Technologies

D1.7.1 Ontology generation from scratch

Blaž Fortuna
Dunja Mladenić, Marko Grobelnik
(Jožef Stefan Institute)

Abstract

When working with large corpora of documents it is hard to comprehend and process all the information contained in them. Standard search engines usually rely on word matching and do not take the structure within the corpus into account. We try to overcome that by automatic extraction of topics covered within the documents, visualization of the corpus and semi-automatic construction of topic ontology.

Keyword list: latent semantic indexing, latent semantic analyasis, clustering, k-means, visualisation, multidimensional scaling, semi-automatic topic ontology construction

WP1
Prototype/Report PU
Contractual date of delivery: 30.06.2005
Actual date of delivery: 11.7.2005

CHANGES

N.B. This section to be deleted before submission to the Commission

Version	Date	Author	Changes
1.0	28.12.04	Blaž Fortuna	The first version of this report.
2.0	29.12.04	Dunja Mladenic	Revised version of the report
2.1	10.1.05	Blaz Fortuna	Changes from Mercedes Blázquez review incorporated
3.0	29.12.04	Dunja Mladenic	Updated with semi-automatic ontology construction
3.1	10.07.2005	Blaz Fortuna	Update with a protptype description for ontology construction
3.2	11.07.2005	Dunja Mladenic	Revised report

SEKT Consortium

This document is part of a research project partially funded by the IST Programme of the Commission of the European Communities as project number IST-2003-506826.

British Telecommunications plc.
Orion 5/12, Adastral Park
Ipswich IP5 3RE
UK
Tel: +44 1473 609583, Fax: +44 1473 609832
Contact person: John Davies
E-mail: john.nj.davies@bt.com

Empolis GmbH
Europaallee 10
67657 Kaiserslautern
Germany
Tel: +49 631 303 5540
Fax: +49 631 303 5507
Contact person: Ralph Traphöner
E-mail: ralph.traphoener@empolis.com

Jozef Stefan Institute
Jamova 39
1000 Ljubljana
Slovenia
Tel: +386 1 4773 778, Fax: +386 1 4251 038
Contact person: Marko Grobelnik
E-mail: marko.grobelnik@ijs.si

University of Karlsruhe, Institute AIFB
Englerstr. 28
D-76128 Karlsruhe
Germany
Tel: +49 721 608 6592
Fax: +49 721 608 6580
Contact person: York Sure
E-mail: sure@aifb.uni-karlsruhe.de

University of Sheffield
Department of Computer Science
Regent Court, 211 Portobello St.
Sheffield S1 4DP
UK
Tel: +44 114 222 1891
Fax: +44 114 222 1810
Contact person: Hamish Cunningham
E-mail: hamish@dcs.shef.ac.uk

University of Innsbruck
Institute of Computer Science
Techikerstraße 13
6020 Innsbruck
Austria
Tel: +43 512 507 6475
Fax: +43 512 507 9872
Contact person: Jos de Bruijn
E-mail: jos.de-bruijn@deri.ie

Intelligent Software Components S.A.
Pedro de Valdivia, 10
28006
Madrid
Spain
Tel: +34 913 349 797
Fax: +49 34 913 349 799
Contact person: Richard Benjamins
E-mail: rbenjamins@isoco.com

Kea-pro GmbH
Tal
6464 Springen
Switzerland
Tel: +41 41 879 00
Fax: 41 41 879 00 13
Contact person: Tom Bösser
E-mail: tb@keapro.net

Ontoprise GmbH
Amalienbadstr. 36
76227 Karlsruhe
Germany
Tel: +49 721 50980912
Fax: +49 721 50980911
Contact person: Hans-Peter Schnurr
E-mail: schnurr@ontoprise.de

Sirma AI EAD, Ontotext Lab
135 Tsarigradsko Shose
Sofia 1784
Bulgaria
Tel: +359 2 9768 303, Fax: +359 2 9768 311
Contact person: Atanas Kiryakov
E-mail: naso@sirma.bg

Vrije Universiteit Amsterdam (VUA)
Department of Computer Sciences
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
Tel: +31 20 444 7731, Fax: +31 84 221 4294
Contact person: Frank van Harmelen
E-mail: frank.van.harmelen@cs.vu.nl

Universitat Autonoma de Barcelona
Edifici B, Campus de la UAB
08193 Bellaterra (Cerdanyola del Vall` es)
Barcelona
Spain
Tel: +34 93 581 22 35, Fax: +34 93 581 29 88
Contact person: Pompeu Casanovas Romeu
E-mail: pompeu.casanovas@uab.es

Executive Summary

When working with large corpora of documents it is hard to comprehend and process all the information contained in them. Standard search engines usually rely on word matching and do not take the structure within the corpus into account. We try to overcome that by automatic extraction of topics covered within the documents, visualization of the corpus and semi-automatic construction of topic ontology.

Contents

1. Introduction

When working with large corpora of documents it is hard to comprehend and process all the information contained in them. Standard text mining and information retrieval techniques usually rely on word matching and do not take into account the similarity of words and the structure documents within the corpus. We try to overcome that by automatic extraction of topics covered within the documents, visualization of the corpus and semi-automatic construction of topic ontology.

We chose two different approaches for discovering topics within the corpora. The first approach is a linear dimensionality reduction technique know as Latent Semantic Indexing (LSI). This technique relies on the fact that words related to the same topic co-occur together more often than words describing different topics. The result of LSI are fuzzy clusters of words each describing one topic. We used this approach as part of dimensionality reduction for visualization and for suggesting topics at semi-automatic construction of topic ontology.

The second approach we used for extracting topics is a well known *k*-means clustering algorithm. It partitions the corpus into *k* clusters so that two documents within the same cluster more closely related that two documents from different clusters. We used this algorithm for suggesting topics at semi-automatic construction of topic ontology.

Visualization of a document corpus is a very useful tool which helps user at discovering the main topics that the documents from this corpus talk about. It allows user at getting to know the corpus better and this can eventually help him when working on topic ontology construction.

Topic ontology is a set of topics connected with different types of relations. Construction of topic ontology based on a given corpus can be a very time consuming task for the user. In order to get a feeling on what are the topics in the corpus, what are the relations between topics and at the end to assign each document to some certain topics, the user has to read all the documents. We overcame this by building a special tool which helps user by suggesting the topics and visualizing the topic ontology created so far – all in real time. This tool in combination with visualization tool aims at assisting the user in semi-automatically creating a topic ontology from a document collection.

2. Summary of existing work

2.1. Representation of text documents

In order to use the algorithms proposed above we must first represent the text documents as vectors. We use standard *Bag-of-Words* (BOW) approach together with IDF weighting [6]. In the BOW representation there is a dimension for each word; a document is then encoded as a feature vector with word frequencies as elements. Elements of vectors are weighted with IDF weights. All *i*-th elements are multiplied with $IDF_i = log(N/df_i)$, where N is total number of documents and df_i is document frequency of *i*-th word (number of documents in which *i*-th word appears). This kind of vectors are also called *TFIDF* vectors. This representation is often referred to as vector-space model. Note that the vector for a specific document is usually very sparse, that is many of its components are zero. This occurs because in a usual text document (news article, web site…) only a small portion of the vocabulary from the whole corpus is used. It is important to take this sparsity into account when dealing

with large corpora since it can significantly influence the speed and the memory requirements of the system.
Different metrics can be defined on text documents once they are represented as vectors. The most widely used one is *cosine similarity*. There the similarity between two documents is defined as the cosine of the angle between their vector representations. More formally, let $\boldsymbol{x}$ and $\boldsymbol{y}$ be vector representations of two text documents. Then the cosine similarity is simply an inner product, $\boldsymbol{x}^T\boldsymbol{y}$, given that the vectors are normalized, that is $\boldsymbol{x}^T\boldsymbol{x} = 1$ and $\boldsymbol{y}^T\boldsymbol{y} = 1$. We will use this metric for comparing text documents. Note that the cosine similarity between two exactly the same documents is 1 and 0 between two documents that share no words.

2.2. Discovery of Latent Semantics

Language contains many redundant information, since many words share common or similar meaning. For computer this can be difficult to handle without some additional information – background knowledge. Latent Semantic Indexing (LSI) is a technique for extracting this background knowledge from text documents. It uses a technique from linear algebra called Singular Value Decomposition (SVD) and bag-of-words representation of text documents for extracting words with similar meanings. This can also be viewed as extraction of hidden semantic concepts or topics from text documents.
Most well known and used approach is Latent Semantic Indexing as described in [1]. First term-document matrix A is constructed from a given set of text documents. This is a matrix with bag-of-words vectors of documents as columns. This matrix is decomposed using singular value decomposition so that $A = USV^T$ where matrices U and V are orthogonal and S is a diagonal matrix with ordered singular values on the diagonal. Columns of matrix U form an orthogonal basis of a subspace in bag-of-words space where vectors with higher singular values carry more information (this follows from theorem that by truncating singular values to only biggest k we get the best approximation for matrix A with rank k). Because of all this, vectors that form the basis can also be viewed as concepts or topics. The space spanned by these vectors is called *Semantic Space*.
Each basis vector is living in bag-of-words space so the elements of this vector can be seen as weights assigned to the words. Geometrically each basis vector splits the bag-of-words space into two halves. By taking just the words with the highest positive or the highest negative weight in this basis vector we get a set of words which best describe a concept generated by this vector. Note that each vector can generate two concepts; one is generated by positive weights and one by negative weights.
Another approach for extracting latent semantics from text documents is Probabilistic Latent Semantic Analysis (PLSA) introduced in [2]. Compared to standard Latent Semantic Analysis which stems from linear algebra and performs a Singular Value Decomposition of co-occurrence tables, this method is based on a mixture decomposition derived from a latent class model. This method assigns each word a probability to be in a concept. Number of concepts is predefined.

2.3. K-Means clustering

Clustering is a technique for partitioning data so each partition (or cluster) contains only points which are similar according to some predefined metric. In the case of text this can be seen as finding groups of similar documents, that is documents which share similar words.

K-Means [7] is an iterative algorithm which partitions the data into *k* clusters. It has already been successfully used on text documents [9] to cluster a large document corpus based on the document topic and incorporated in an approach for visualizing a large document collection [8]. The algorithm is roughly as follows:

Input: A set of data points, a distance metric, the desired number of clusters *k*
Output: Clustering of the data points into *k* clusters
Set *k* cluster centers by randomly picking *k* data points as cluster centers
Repeat
  Assign each point to the nearest cluster center
  Recompute the new cluster centers
Until the assignment of data points has not changed

2.4. Visualization

Visualization of a set of text documents is a very useful tool for finding the main topics that the documents from this set talk about. For example, given a set of descriptions of European research projects in 6th Framework IST, one can find main areas that these projects cover, such as, semantic web, e-learning, security, etc. Bag-of-words representation of text has very high dimensionality, so in order to represent text documents in 3D world number of dimensions has to be reduced. This can be done by first extracting main concepts from documents with LSI and then using this information to position documents on two dimensional plane that can be plotted on computer screen.

Many methods were developed for visualizing text documents or high dimensional data in general. Some examples are *Themeview, Themeriver, Topic Islands* (http://www.pnl.gov/infoviz), *Self-Organizing maps* (http://websom.hut.fi/websom/), Graph and Tiling visualization [8].

In this work we focused on linear subspace methods and multidimensional scaling. They can be both applied to any data set given as vectors in some higher dimensional vector space. We focus on methods for reducing the number of dimensions to two so data set can be shown on computer screen.

Linear subspace methods, like Principal Component Analysis (PCA) [3] or Latent Semantic Indexing, focus on finding direction in original vector space, so they capture the most variance (PCA) or are the best approximation for original document-term matrix (LSI). By projecting data (text documents) only on first two directions we get points which live in two dimensional space.

The problem with this approach is that only the information from first two directions is preserved. In case of LSI it would mean that all documents are described using only two main concepts or topics!

Another approach is called multidimensional scaling [4]. At this method, points, representing documents, are positioned into two dimensions so they minimize some energy function. The most common form of this function is

$$E = \sum_{i \neq j} (\delta_{ij} - d(x_i, x_j))^2 , \qquad (1)$$

where x_i are two dimensional points and δ_{ij} represents the similarity between documents i and j. An intuitive description of this energy function is: the better distance between the points on plane approximate the real similarity between documents, the lower the energy is. Function E is always nonnegative and is zero only when distance between points matches exactly the similarity between documents. Another version of energy function is

$$E = \sum_{i \neq j} \frac{(\delta_{ij} - d(x_i, x_j))^2}{d(x_i, x_j)^2}. \quad (2)$$

Here the distances between points belonging to similar documents have higher weights. On this way distances between points are locally better approximations for real similarity between documents.

3. Description of the approach

3.1. Topic discovery

We chose to use LSI and *k*-means for the discovery of topics. Both approaches are well studied and were already successfully used in many applications like information retrieval, cross-lingual text mining, etc and can be very efficiently implemented so they are scalable to the size of the corpora and the dimensionality of the bag-of-words space.

As we mentioned in section 2.2 a linear algebra technique called SVD is used to compute LSI. This method is well studied in areas of numerical linear algebra and efficient algorithms for calculating singular decomposition exist. The basis of LSI is the calculation of singular decomposition of term-document matrix *A*. This matrix is usually very sparse and of large dimensions. We chose Lanczos algorithm since it has exactly the properties we seek for:

- no transformation is done on matrix *A* (this is crucial since matrix *A* is very large and very sparse and any transformation which could ruin this sparsity would require to much memory for storing the matrix),
- we can take advantage of sparseness of matrix *A* (i.e., very fast multiplication of matrix *A* with vectors),
- it is iterative – at each iteration we get another singular tripled, first one with highest singular value, then one with second largest singular value and so on,
- is scalable to very large number of documents or words.

More information on this algorithm can be found in [5].

The implementation of *k*-means algorithm is also very efficient and it scales logarithmically to the number of documents.

3.2. Visualization

Since both methods (linear subspace and multidimensional scaling) have some nice properties we choose a combination. This is the algorithm:

	Input: Set of documents to visualize in form of TFIDF vectors **Output**: Set of two dimensional points representing documents
1)	Calculate *k* dimensional semantic space generated by input set of documents, this is done using LSI
2)	Project documents onto the semantic space
3)	Apply multidimensional scaling using energy function *(1)* on documents with Euclidian distance in semantic space as similarity measure

There are two main problems to be solved so the above algorithm works. First problem is how to determine the value of *k*. This can be done by checking the singular values. Let $\Sigma_k = S_1 + S_2 + \ldots + S_k$. We know that $\Sigma_n = Trace(A^T A)$, where *n* is the number of documents and *A* is the term-document matrix. From this we can calculate *k* by prescribing the ratio Σ_k/Σ_n to some fixed value, for example 50 %.

A more difficult problem is how to perform multidimensional scaling efficiently. One way is to use gradient descent. The problem with this approach is that the energy function is not convex and it has many local minima which are not interesting for us. One could start this method more times with different initial state and then choose results with the lowest energy. This energy function can also be reformulated in the following way. Given some position of points we calculate how to move each point so we minimize energy function. Lets denote with (x_i,y_i) current position of points and with $(x_i',y_i')=(x_i+\Delta x_i, y_i+\Delta y_i)$ desired position. Then we have

$$\begin{aligned} \tfrac{1}{2}(d_{ij}'^2 - d_{ij}^2) &= \tfrac{1}{2}[(x_i + \Delta x_i - x_j - \Delta x_j)^2 + (y_i + \Delta y_i - y_j - \Delta y_j)^2 - (x_i - x_j)^2 - (y_i - y_j)^2] \approx \\ &\approx (x_i - x_j)\,\Delta x_i + (x_j - x_i)\,\Delta x_j + (y_i - y_j)\,\Delta y_i + (y_j - y_i)\,\Delta y_j = \\ &= [(x_i - x_j), (x_j - x_i), (y_i - y_j), (y_j - y_i)][\Delta x_i, \Delta x_j, \Delta y_i, \Delta y_j]^T. \end{aligned}$$

By writing this for each pair (i,j) and substituding d_{ij}' with the original distance between i-th and j-th document we get a system of linear equations which has a vector of moves (Δx and Δy) for a solutions. This is an iteration which finds a step towards minimizing energy function and is more successful at avoiding local minima. Each iteration involves solving a linear system of equations with a very sparse matrix. This can be done very efficiently using Conjugate Gradient (CG) method.
At the end, points are normalized so they lie in square $K = [0,1]^2$.

3.2.1. Visualization beyond dimensionality reduction

After text-documents are mapped onto plane other techniques can be used to make the structure of documents more explicit for users:

- **Landscape generation**: the landscape can be generated by using the density of points. Each point from square K is assigned height using the formula
$$h(x,y) = \sum_{i=1}^{n} Exp(-\sigma \,\|\,(x,y) - (x_i,y_i)\,\|^2)\,.$$
The diversity of landscape is controlled by parameter σ.
- **Keywords**: each point from square K can be assigned a set of keywords by averaging TFIDIF vectors of documents which appear within a circle with centre in this point and radius R.

These features can be used when showing visualizations for making them more descriptive.

3.3. Semi-automatic construction of topic ontology

We view semi-automatic topic ontology construction as a process where the user is actually doing the construction and taking all the decisions while the computer only gives suggestions for the topics, helps by automatically assigning documents to a topics, etc. After the user finishes the output can be saved as XML or RDF. There are three major parts of the system.

3.3.1. Topic suggestion

When user selects a topic the system automatically suggests what the subtopics could be. This is done by Latent Semantic Indexing or k-means algorithms applied only to the documents from the selected topic. The number of suggested topics is supervised by the user. User then selects the subtopics he finds reasonable and the system adds them to the ontology as subtopics of the selected topic.

3.3.2. Topic management

The user can manually edit each topic which he added to the topic ontology. He can change which documents are assigned to this topic (one document can belong to more topics), what is the name of the topic and what is the relationship of the topic to other the topics. The main relationship is *subtopic-of* and is automatically added when adding subtopics as described in the previous section. The user can control all the relations between topics by adding or removing the relations. The only limit is that when topic *A* is a subtopic of topic *B*, then *B* can not be a subtopic of *A*.

In this step the systems helps user by automatically assigning the documents to a topic when the topic is added to the ontology. The system also has two methods for suggesting the user what are the main keywords describing the topic: (1) keyword extraction using centroid vectos and (2) keyword extraction using SVM. This helps the user to get an idea about the topics and decide how to name the topics.

The first method (1) for keyword extraction is done by using the centroid vector of the topic (centroid is the sum of all the vectors of the document inside the topic). The keywords are selected as the words with the highest weights in the centroid vector.

The second method (2) is based on Support Vector Machine (SVM) binary classifier as follows. Let *A* be the topic which we want to describe by keywords. We take all the documents from the topics that have *A* as a subtopic and mark these documents as negative. We take all the documents from *A* and mark them as positive. If one document is assigned both negative and positive label we say it is positive. Then we learn a linear SVM classifiers on these documents and classify the centroid of the topic *A*. Keywords describing the concept *A* are the words, which weight in SVM normal vector contribute most when deciding if centroid is positive.

The difference between these two approaches is that the second one takes into account the context of a topic. Let's say that we have a topic named "computers". When deciding, what are the keywords for some subtopic *A*, the first method would only look at what are the most important words within in the subtopic and words like "computer" would most probably be found as important. However, we already know that *A* is a subtopic of "computers" and are more interested in what are the keywords that separate it from the other documents within the "computers" topic. The second method does that by taking the documents from all the super-topics of A as a context.

3.3.3. Ontology visualization

While the user is constructing/changing topic ontology, the system visualizes it in a real time as a graph with topics as nodes and relations between topics as edges. User can click on a specific node with the same effect as selecting the topic from a list.

4. Architecture

The whole process of extracting topics, visualization and semi-automatic topic ontology construction is fully integrated into Text Garden text mining library. Figure 1 shows the pipeline for visualization. Utilities from Text Garden related to topic extraction, visualization and topic ontology construction are:

- **Bow2SemSpace** – utility for generating semantic space using LSI. It gets on input the Text Garden Bag-Of-Words file with documents that will be used for extracting latent semantics.
- **BowKMeans** – utility for clustering using *k*-means. It gets on input the Text Garden Bag-Of-Words file with documents and it outputs the partition of this documents into clusters.

- **Bow2VizMap** – utility which calculates two dimensional representation of documents using projection on semantic space and multidimensional scaling. It gets Bag-Of-Words file containing documents for visualization and the output from *Bow2SemSpace* as an input.
- **Document Atlas** – utility for showing and exploring visualizations. Documents are presented as points on a map and density is shown as a texture in the background. Most common keywords are shown for each area of map. When the user moves mouse around the map a set of most common keywords is shown for the area around the mouse (area is marked with a circle). The user can also zoom in to see specific areas in more details. By clicking on a document, more information about it is shown on the left side of the screen. Example of how this utility looks like is shown in Figure 2.
- **OntoGUI** *(code name)* – utility for interactive topic ontology construction. It gets as input the Text Garden Bag-Of-Words file. It allows user to created topics, organize the topics into topic ontology and assign the documents into topics. It also uses machine learning techniques described in previous sections to help user at each step: it gives suggestions on what the topics are, how they are related, gives suggestions on the name of the topics and can automatically assign the documents into topics.

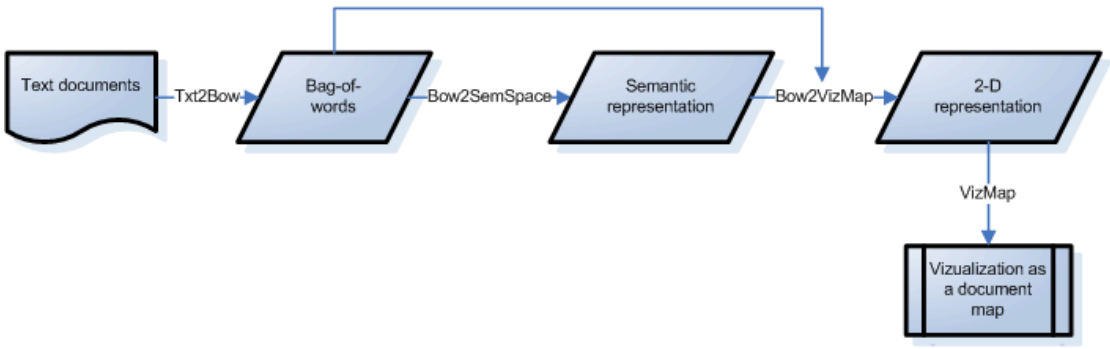

Figure 1. *Pipeline of utilities for visualization in Text Garden.*

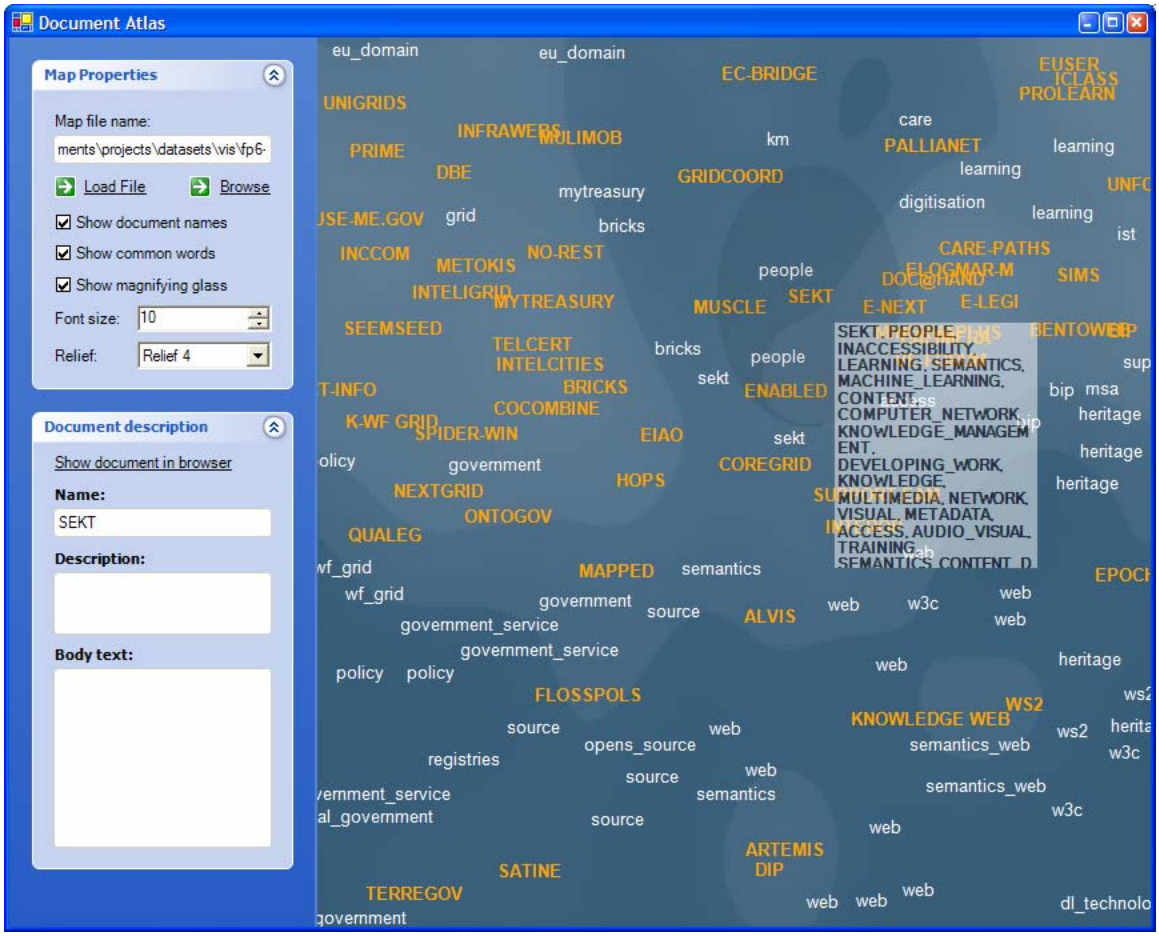

Figure 2. *Visualization of IST project with zoom on semantic web and knowledge technologies projects.*

For more information on how to use these utilities see Appendix – User Guide.

5. Future development

We envision that more work will be done on how to extract as many information as possible from corpus of documents, how to take full use of the output of LSI and k-means, to improve them and maybe add other techniques for concept/topic discovery.

We already used our visualization approach for different scenarios: for visualizing projects, scientific articles, companies, movies, customers, etc. We will use further user feedback to add new features which would make this tool even more informative and useful. One area not fully explored is the use of background landscape in visualization. Now we use it to show the density of documents but it can serve as well to show some other attributes. Document Atlas together with visualization could also be used as an user interface for some machine learning algorithms like active learning, clustering, etc.

There is also a large possible area of improvements to semi-automatic ontology construction tool. The most important next step is to try it in some practical scenarios and see how it fits the needs of the users and what features should we add. Another direction is towards making the whole process more automatic so even less user interaction will be needed. This involves things like calculating the quality of topics suggested by the system, more automated discovery of the optimal number of topics, etc. It would also be interesting to try some other methods for topic discovery, for example PLSA.

6. Conclusions

We studied and implemented methods for discovering concepts or topics within corpus of documents and for visualizing the corpus and combined them on a novel way into a visualization system which lets the user to explore the document space. We also designed and fully implemented a system for semi-automatic construction of topic ontology based on a corpus of documents.

7. Appendix – User Guide

Utilities for semantic space learning (LSI) and visualization are part of pipeline in Text Garden and are fully compatible with other utilizes (see Figure 1). Bow2SemSpace and Bow2VizMap are command line utilities that take files in Text Garden formats as input and output files in Text Garden format. Document Atlas is the user interface for exploring visualizations and it runs as a windows application.

7.1. Bag-Of-Words to Semantic-Space

The utility learns semantic space from documents stored in Bag-Of-Words input file ("-i"). Semantic space is then stored as binary ".ssp" file ("-ob") or as text file ("-ot"). Binary output can then be used as input for other utilities and text output shows most common words for each basis vector of semantic space.

The parameter "-t" determines which method will be used for generating semantic space. The parameter "-dims" determines the dimensionality of semantic space. The parameter "-reorto" is specific to LSI and determines what kind of reortogonalization will be performed: "none" is the fastest but also least numerical stable; "selective" is in the middle and "full" is the slowest but most stable. When number of documents in input file is small use "full", otherwise "selective" should work fine. Use "none" only when time is really crucial and both "full" and "selective" are not fast enough.

usage: `Bow2SemSpace.exe`

`-i:`	Input-BagOfWords-File (default:")
`-ob:`	Output-SemanticSpace-Binary-File (default:")
`-ot:`	Output-SemanticSpace-Text-File (default:")
`-t:`	Semantic-Space-Type (lsi, pca) (default:'lsi')
`-dims:`	Number-Of-Space-Dimensions (default:50)
`-reorto:`	Reortogonalization (none, selective, full) (default:'selective')

Example: `Bow2SemSpace.exe -i:fp6.bow -ob:fp6.ssp -ot:fp6.txt -t:lsi -dims:30 -reorto:full`

The above example learns semantic space with 30 dimensions from set of documents from Bag-Of-Words file fp6-ist.bow using LSI with full reortogonalization.

7.2. Bag-Of-Words k-Means

The utility performs K-Means clustering procedure on the input file ("-i") in the Bag-Of-Words format ".Bow". It produces 3 types of output files:

1. coded file with the partition ("-op"),
2. text file with the description of the clusters, centroids and quality measures,
3. XML file with the result of clustering.

With the parameter "-docs" the number of clustered documents is determined (value "-1" means all documents"). The parameter "-clusts" determines the final number of clusters. The parameter "-rseed" determines the value of random-number-generator seed, where value 0 means nondeterministic value. The parameter "-ctrials"

determines the number of different runs/trials of K-Means algorithm in a search for the best solution. The parameter "-ceps" determines convergence epsilon value which influences the stopping criterium for the K-Means algorithm. The parameter "-cutww" determines the percentage of the sum of the weights for the best words in the centroids which appear in the textual output file. The parameter "-mnwfq" determines the minimal document-frequency of the words which are used for the document representation.

usage: **`BowKMeans.exe`**

`-i:`	Input-File (default:")
`-op:`	Output-BowPartition-File (default:'KMeans.BowPart')
`-ot:`	Output-Txt-File (default:'KMeans.Txt')
`-ox:`	Output-Xml-File (default:'KMeans.Xml')
`-docs:`	Documents (default:-1)
`-clusts:`	Clusters (default:10)
`-rseed:`	RNG-Seed (default:1)
`-ctrials:`	Clustering-Trials (default:1)
`-ceps:`	Convergence-Epsilon (default:10)
`-cutww:`	Cut-Word-Weight-Sum-Percentage (default:0.5)
`-mnwfq:`	Minimal-Word-Frequency (default:5)

Example: **`BowKMeans.exe -i:Reuters21578.Bow -docs:1000 -clusts:10`**
The above example call clusters first 1000 documents (-docs:) from Reuters21578.Bow (-i:) into 10 clusters (-clusts:). Files KMeans.Txt (textual description of results), KMeans.Xml (results in XML form) and KMeans.BowPart (binary representation of partition) are created.

7.3. Bag-Of-Words to Visualization-Map

The utility visualizes documents stored in Bag-Of-Words input file ("-i"). Calculated positions of documents is then stored in .viz file ("-o"). Semantic space can be used to help at lowering dimensions ("-ssp"). Extra landscape maps can also be calculated on already calculated positions of documents ("-viz").
The parameter "-ssp_dim" determines to how many dimensions input documents should be reduced using semantic space (if semantic space is given as input). The parameters "-max_step" and "-min_diff" determine how accurate visualisation will be. The higher the "-max_step" parameter and lower "-min_diff" parameter, the more exact it will be but it will also take longer to compute.
The parameter "-numrlf" determines number of landscape maps that will be calculated. The parameters "-rlfx" and "-rlfy" determine the resolution of calculated landscapes. The parameters "-sigma" and "-k" determine the diversity of landscapes.

usage: **`Bow2VizMap.exe`**

`-i:`	Input-BagOfWords-FileName (default:")
`-ssp:`	Input-Sematic-Space-FileName (default:")
`-viz:`	Input-Sematic-Space-FileName (default:")
`-o:`	Output-Points-FileName (default:")
`-ssp_dim:`	Reduced-Space-Dimension (default:50)
`-max_step:`	Max-Number-Of-Iteration (default:1000)
`-min_diff:`	Min-Iteration-Difference (default:0.0001)
`-numrlf:`	Number-Of-Landscapes (default:4)

`-rlfx:`	Width-Of-Landscapes (default:1000)
`-rlfy:`	Height-Of- Landscapes (default:1000)
`-sigma:`	Sigma-Parameter-For-RBF-Kernel (default:0.03)
`-k:`	Sigma-Decreasing-Coficient (default:0.5)

Example: **`Bow2VizMap.exe -i:fp6-ist.bow -ssp:fp6-ist.ssp -o:viz.viz -ssp_dim:100 -max_step:2000 -numrlf:10 -simga:0.05`**

The above examples visualize documents from fp6-ist.bow by first projecting them to semantic space of 100 dimensions. It can do at maximal 2000 iterations and 10 landscapes are pre-calculated.

7.4. Document Atlas

Document Atlas is a utility for exploring visualizations created by Bow2VizMap. This utility is controlled trough top panel in side bar (Figure 3). In order to load visualization from disk, write its file name in "Map file name" text box or use "browse" link. Then click "Load File" link. View of visualization can be controlled using other controls:

- *Show document names* – determines, if text documents are presented with full name or just with cross,
- *Show common words* – determines, if common words are shown for regions of the map,
- *Show magnifying glass* – determines, if list of most common keywords is shown next to mouse pointer,
- *Font size* – size of fonts on the map,
- *Relief* – which landscape to use for background.

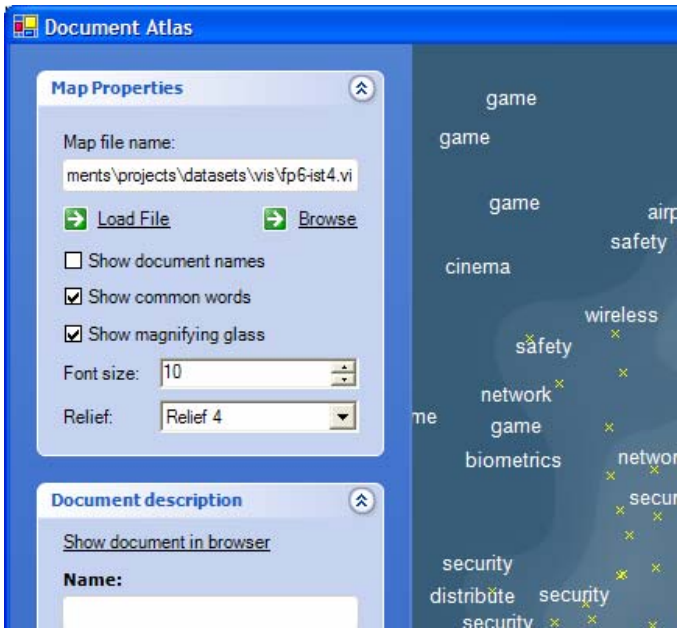

Figure 3. *sidebar for selection options of visualization*

Zooming of the visualization is done by drawing a square around desired area. Right mouse button is used for zoom out. Mouse wheel is used to select the size of are covered with magnifying glass. By clicking on a project more details are shown in the second panel in side bar. By clicking link "Show document in browser" document's url is opened in default browser.

7.5. OntoGUI

OntoGUI is a tool which helps the user at constructing topic ontology. User first loads the Text Garden Bag-of-Words file by clicking "Load BOW" button. The window is split into three areas (see Figure 4). On the left is the list of all the topics use created so far and suggestions for new topics. In the middle is the visualization of the ontology and at the bottom are the properties of currently selected topic. This is only show when "Show detail" is selected next to the list of the topic.

The left side is the area where user can create new topics, merge two topics or delete a topic. A list of all created topics is in the top left list. This is the place where user can select a topic for editing or exploring its subtopics. User can create an empty topic by clicking "New" button; a topic can be deleted by first selecting it and then clicking "Delete" button; two topics can be merged by selecting both (using *ctrl* key) and then clicking "Unite" button. Lower is the list of all the subtopics of the currently selected topic. At the bottom is the list with suggestions for the new subtopics of the currently selected topic. User can apply suggestions by clicking the checkbox next to the suggested topic ID. By clicking "Add concepts" the selected suggestions are added as subtopics and by clicking "Break concept" the currently selected topic is replaced by the selected suggestions.

The visualization area adjusts to the user actions in real time. User can also select a certain topic by clicking its node in the visualization (see Figure 5).

The bottom area is used to modify the properties of the selected topic. The name of the topic can be changed (system helps by showing the most informative words of the topic), the relations of the selected topic towards other topics can be managed and the documents can be assigned to the selected topic.

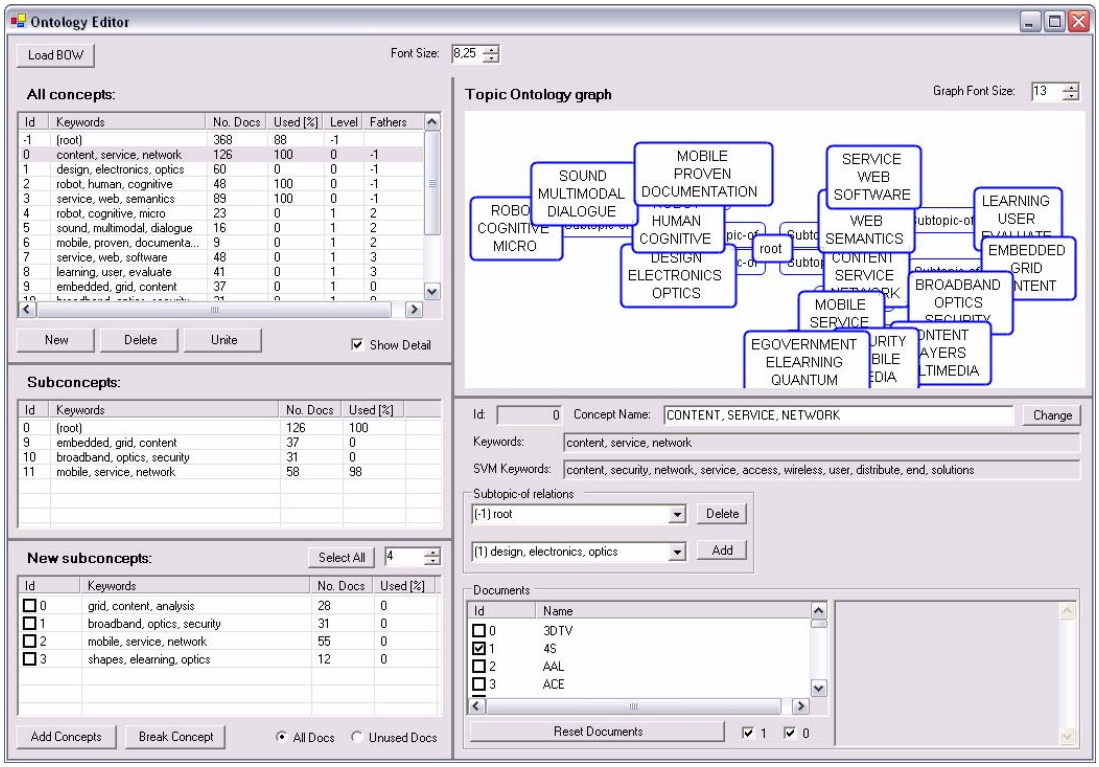

Figure 4. *OntoGUI's main interface*

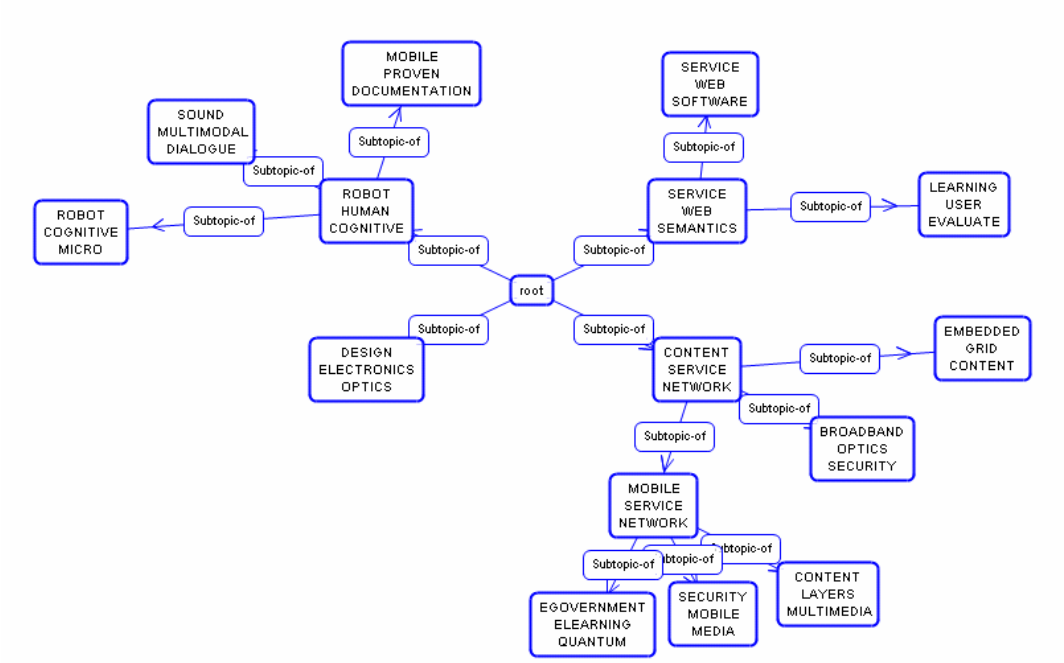

Figure 5. *Visualization of topic ontology*

8. Bibliography and references

1. S. Deerwester, S. Dumais, G. Furnas, T. Landuer and R. Harshman, *Indexing by Latent Semantic Analysis,* Journal of the American Society of Information Science, vol. 41, no. 6, 391-407, 1990
2. Thomas Hoffman*, Probabilistic Latent Semantic Analysis,* Proc. of Uncertainty in Artificial Intelligence, UAI'99, 1999
3. John Shawe-Taylor, Nello Cristianini, *Kernel Methods for Pattern Analysis,* Cambridge University Press, 2004, 143-150
4. Carroll, J.D. and Arabie, P. *Multidimensional scaling*. In M.R. Rosenzweig and L.W. Porter(Eds.). *Annual Review of Psychology*, 1980, *31*, 607-649.
5. Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems*
6. G.Salton. *Developments in Automatic Text Retrieval*, Science, Vol 253, pages 974-979, 1991
7. Jain, Murty and Flynn: *Data Clustering: A Review*, ACM Comp. Surv., 1999
8. Grobelnik, M., and Mladenic, D.: *Efficient visualization of large text corpora*. Proceedings of the Seventh TELRI seminar. Dubrovnik, Croatia, 2002
9. Steinbach, M., Karypis, G., Kumar, V.: *A comparison of document clustering techniques*. In Proceedings of KDD Workshop on Text Mining, pp. 109–110, 2000