Content Visualization of Scientific Corpora Using an Extensible Relational Database Implementation

Theodoros Giannakopoulos(✉), Eleftherios Stamatogiannakis, Ioannis Foufoulas, Harry Dimitropoulos, Natalia Manola, and Yannis Ioannidis

Management of Data, Information, and Knowledge Group,
Department of Informatics and Telecommunications,
University of Athens, Athens, Greece
tyiannak@di.uoa.gr
http://www.madgik.di.uoa.gr/

Abstract. A method for supervised classification and visualization of collections of scientific publications is presented. By integrating a text classification module, which leads to class probability estimation, along with a dimensionality reduction technique, which represents each class in the 2-D space, any collection of unlabelled documents can be visualized. The classification and visualization modules have been trained on three different datasets and respective categorizations. We provide an example of our system's functionality by visualizing the content of collections of publications which share a common funding scheme. In order to implement this, we have developed a funding mining submodule which identifies documents of particular funding schemes. All the individual modules have been implemented using the madIS system, which provides data analysis functionalities via an extended relational database.

1 Introduction

Visualization of textual content is a rather important task in data mining, since it constitutes a way to represent what particular document corpora refer to. Among the wide range of textual information available in the web, scientific documents form a subject of major interest, making the text analytics task for such content more complex and interesting, mainly due to their content richness and diversity. In this work we present a method for supervised classification and visualization for collections of scientific publications. The classification and visualization modules have been trained using a series of taxonomies and respective datasets (e.g., the arXiv dataset [1]). In addition, special focus has been given on the integration of both the classification and visualization modalities: in particular this is achieved in the context of a data processing workflow. We demonstrate intermediate results of this procedure, applied on a particular use case, according to which we are interested in visualizing the content of collections of publications which *share a common funding scheme* (e.g. FP7-ICT). Towards this end,

Ł. Bolikowski et al. (Eds.): TPDL 2013, CCIS 416, pp. 101–112, 2014.
DOI: 10.1007/978-3-319-08425-1_10,

we have also implemented a fast and accurate fund mining submodule which is responsible for detecting funding information in scientific publications.

The proposed system has been implemented under the `OpenAIREplus` EU project titled: "2nd-Generation Open Access Infrastructure for Research in Europe" (283595)[1]. This project aims to build an information infrastructure of scientific publication and data repositories. In addition, it implements EC's *open access* policies through services that effectively connect publications to scientific data and funding information. This is achieved through data mining techniques towards the automatic classification, indexing and clustering of scientific publications, funding information mining and content visualization. In the context of such an infrastructure, content classification and visualization can be a rather powerful tool for research administrators, if such techniques are connected to funding schemes. In other words, automatic content visualization can assist the research administrators in the process of strategy or policy making, since it will provide them with "an image" of the content distribution for particular funding schemes or any other categorization of interest (e.g., by country, institutes, etc.).

2 Methodology

2.1 Content-Based Classification

The purpose of this submodule is twofold: (a) To provide a simple way of text classification: towards this end a rather straightforward technique is adopted. (b) To represent the content of each document class. This representation will be also used in the visualization step.

Feature Extraction. The feature extraction is rather straightforward, since it only involves the extraction of term frequencies, which will be later used to estimate the respective probabilities used in the classifier. Towards this end, some typical text mining steps are adopted. In particular, the following steps are executed for each document d:

1. Tokenization: the initial stream text is broken into individual words. Also, a separate process for detecting frequent word 2-grams is used.
2. Stop word removal: towards this end, we have adopted the NLTK's stop words list [2]
3. Stemming: for this preprocessing step the Wordnet lemmatizer of NLTK has been adopted [2]
4. A list of unique terms is generated after the first preprocessing steps
5. For each unique term t its frequency $df_d(t)$ is extracted

As the above process continues for a corpus of documents, a global list of unique terms is updated, along with the respective term frequencies for each document. By "terms" we mean single words or 2-grams, as explained above.

[1] http://www.openaire.eu/en/component/content/article/326-openaireplus-press-release

Classification. Since we are interest in a supervised learning scheme we obviously make the assumption that each document is mapped to one or more classes. There are several of such available datasets (and respective taxonomies) which will be discussed later. In the context of a supervised task, we execute for each class c the process described in Sect. 2.1 in order to compute the list of terms and the corresponding frequencies for each document in the corpus (df_d). Then, based on these term frequencies, the following probabilities are estimated (a) $P(t)$: the a-priori probability that term t can appear in the whole corpus and (b) $P(t|c)$: the probability that term t appears in some document of a particular class c.

These probabilities are used to produce a set of dictionaries (D_c) and respective weight arrays (W_c) for each class c. In particular, for each class c and for each term t in the list of corpus terms, if $\frac{P(t|c)}{P(t)} > T$, where T is a user-defined threshold, then the term t is added to dictionary D_c. Also, the respective weight vector (W_c) is updated according to: $W_c(t) = \frac{P(t|c)}{P(t)}$. In this way, each class obtains a dictionary-based representation which is used both for classification and visualization. In the case of document classification, for the document d, with terms $t_1, \ldots, t_N$ the probability

$$P_d(c) = \sum_{i=1}^{N} log(W_c(j : D_c(j) = t_i))$$

is computed. This classification step is actually equivalent to the Naive Bayes classifier [3].

2.2 Content Visualization of Text Corpora

Class Numerical Representation and Dimensionality Reduction. As explained in the previous Section, each content class is represented through a pair of term dictionaries and respective weights. Our final purpose is to represent a corpus of scientific publications in a human perceptible dimensionality. In particular, the visualization module aims to map each of the content classes (through their dictionary representation) to a particular point in the 2D space, so that similar classes (in terms of textual content) are close to each other. This is actually a dimensionality reduction task.

Let us first create a numerical representation of the content classes. Towards this end, a similarity-between-classes matrix is extracted. This matrix is defined based on the extracted class dictionaries and respective weight arrays (Sect. 2.1). In particular, let c_1 and c_2 be a pair of content classes and D_{c_1}, D_{c_2}, W_{c_1} and W_{c_2} the related dictionaries and weights respectively. N_1 and N_2 are the number of dictionary terms for the two classes. The transition from the "dictionary space" to a purely numerical representation is achieved through the similarity measurement calculation:

$$S(c_1, c_2) = \frac{\sum_{i=1}^{N_1} W_{c_2}(k : D_{c_1}(i) = D_{c_2}(k))}{\sum_{i=1}^{N_2} W_{c_2}(i)} + \frac{\sum_{i=1}^{N_2} W_{c_1}(k : D_{c_2}(i) = D_{c_1}(k))}{\sum_{i=1}^{N_1} W_{c_1}(i)}$$

In other words, we directly use the extracted dictionaries to calculate the similarity between any pair of classes. An alternative of this similarity technique would be some type of sample-to-sample similarity measurement. However, the extraction of the adopted class-based similarity matrix is of low computational cost, which makes it attractive in the context of a system that processes very large corpora. In addition, this approach makes direct use of the information extracted in the training phase of the content classifier.

Using matrix S we have managed to numerically represent the class distributions in the $\mathbb{R}^M$ space, where M is the total number of classes, from a simple dictionary representation. Now, we need to find a combination of the dimensions of this space to a 2D space which will provide us with the visualization results. However, since M can be quite high for most of the taxonomies (e.g. the arXiv dataset has almost 150 classes) the dimensionality task cannot be achieved directly, due to numerical issues. So instead of proceeding with the final dimensionality reduction procedure, we firstly reduce the feature space via a clustering technique. In particular, we first apply a k-means clustering algorithm [4] on S. Towards this end, we assume rows are samples. k is selected to be equal to 20. Therefore, this clustering procedure leads to a set of k cluster centers, say $Cl_i, i = 1, \cdots, k$. This clustering is a grouping of similar content classes, based to the dictionary-based similarity criterion. As a next step, we compute the Euclidean distance between each row i of the S matrix and each cluster j: $d(i, j)$. d can be assumed as a set of M samples in the k feature space.

The clustering procedure described above leads to a new numerical (lower dimension) representation of each class based on its distances from the extracted content clusters. A self organizing map (SOM) is then used in order to extract a 2-D discretized representation of the classes [5], based on that lower dimensional space. Now each class, is represented by a pair of discretized coordinates in the 2-D feature space. The reason why we selected to express the original feature space as a linear combination of distances from cluster centers, before proceeding with the main SOM algorithm, is to avoid numerical and computational issues that stem from the high initial dimensionality and the large number of classes involved in the SOM training procedure. Reducing the class dimensionality to $k = 20$ makes the training of the SOM faster and more accurate.

Content Visualization of Unlabelled Corpora. The classification-visualization submodules described above classify an unknown document to a set of classes, providing soft outputs (probability estimations) and represent each of the classes in the 2-D space. If these two submodules are combined and applied for a corpus of unknown scientific documents, they can provide a visual interpretation of the corpus content. In more detail, the content classification module is firstly applied for every document $i, i = 1, \ldots, N_d$. N_d is the total number of documents in the collection. This leads to a set of output probabilities $P_i(c)$ for each class $c = 1, \ldots, M$. When the classification stage is completed for each document, the corpus content is represented using a 3-D vector for each class c: $[X_c, Y_c, \frac{\sum_{i=1}^{N_d} P_i(c)}{N_d}]$, where, X_c, and Y_c are the estimated 2-D class

coordinates extracted from the training phase of the visualization module. According to that, each document corpus is represented for each class of the adopted taxonomy, using: (a) the 2-D estimated coordinates and (b) the accumulated (average) estimated content class probability. This can be illustrated in the 2D plane using a "balloon" representation for the visualization of this type of 3-D data, provided by Google.[2]

2.3 Fund Mining

The purpose of the fund mining submodule is to detect documents of particular funding schemes. The initial aim was to identify EU FP7-funded[3] documents, however, recently this was extended to also include Wellcome Trust[4] projects. We have now further enhanced the module so that it is able to handle an arbitrary number of funding bodies and institutions. As explained in the introduction, funding information is a rather important type of metadata that can be useful in article, authors, institutions and funding statistics and visual analytics. Consider for example the task of discovering and presenting trends in a temporal way, or tracking statistics for different funding bodies and calls. In the context of the present work, funding information is used to specify the types of documents being visualized.

The funding mining module can either be used on individual publications (e.g., at the time a publication is deposited to facilitate authors in adding metadata information), or in batch mode (e.g., in order to processes all documents found in a collection/repository). In both cases, the module does some preprocessing (such as stop word removal, tokenization, etc.) on the text of each publication. Then it scans the text to extract possible matches using patterns, and it finds matches against the current known lists of project grant agreement numbers and/or acronyms for various funding bodies. In addition, contextual information is used to provide a confidence value for each match and to filter out any false matches. Specifically, the context is considered as a bag of positioned weighted words/n-grams. Some terms have positive weights (ex. fp7, european research council) whereas some others have negative (ex. postal code, telephone number etc.). The absolute weights may also increase according to their distance from the matching grant id/acronym. For the selection of terms we use a naive bayesian method on the contexts of manually classified matches. Typical examples of false matches are: gene accession numbers, postcodes, report numbers, other identifiers, etc., which may appear identical to valid grant numbers. Matching by acronym only (in publications where a corresponding grant number is not given) is even more prone to false matches, since acronyms can frequently be identical to regular words. Also, publications may reference project grants in a literature survey section or the references section, without actually being funded by those particular projects; these situations must be identified as false matches.

[2] https://developers.google.com/chart/
[3] http://cordis.europa.eu/fp7/
[4] http://www.wellcome.ac.uk/

3 Implementation Issues

All the training submodules (classification and visualization) have been entirely implemented in Python using a wide series of external libraries (e.g. NumPy[5] and NLTK). However, a seperate release version is needed for the testing case so that:

- implementation is achieved in the context of a data processing workflow
- it can be easily transferred to a distributed environment

Since the adopted scheme only involves text segmentation, dictionary terms retrieval, a simple computation of frequency-related weights and a computation of average soft probabilities, the implementation of the testing phase of both the classification and the visualization modules can be achieved in the context of a data processing workflow. Towards this end, we have used the madIS [6] system which provides data analysis functionalities via an extended relational database. madIS is built on top of the SQLite database with several Python extensions and it feels like Hadoop SQL, without the overhead but also without the distributed processing capabilities.

In madIS, queries are expressed in madSQL: this is an SQL-based declarative language extended with UDFs (User Defined Functions). UDFs are also supported by database systems for a long time; however, their use is limited due to their complexity and limitations. One of the goals of madIS is to eliminate the effort of creating and using UDFs by making them a first class citizens in the query language itself. These features combined, make madIS an agile data analysis environment that encourages rapid experimentation. madIS supports the following three types of UDFs:

- *Row functions.* These take as input one or more columns from a row and produce one value (e.g. the UPPER() function). They are analogous to the Map operator of Map/Reduce systems.
- *Aggregate functions.* These can be used to capture arbitrary aggregation functionality beyond the one predefined in SQL (i.e., AVG(), etc.). They are analogous to the Reduce operator of Map/Reduce systems.
- *Virtual table functions.* These are adopted in order to create virtual tables that can be used in a similar way with tables.

Using this UDF functionality along with the traditional database functionalities, and since the UDFs are closely tied to the relational DB engine in madIS, the communication cost between the two execution layers (functional and relational) is eliminated. In this context, the classification module for example, can be implemented by using (a) a UDF to split document into words, (b) the relational facilities to calculate word frequencies and (c) aggregate UDFs to compute sum of logs. Note that the whole process can be achieved in one single madSQL query, completely within madIS. Finally, apart from the testing phases of the classification and visualization modules, the funding mining module has been entirely built on top of the madIS system.

[5] http://www.numpy.org/

4 Datasets and Taxonomies

Both the classification and visualization modules have been trained on three different datasets and respective taxonomies, namely:

- arXiv (https://www.arxiv.org). This is a large archive for electronic preprints of scientific publications, covering a wide range of fields. The arXiv categorization is rather simple: 2 levels of hierarchy with a total of almost 130 class labels in the second level. These 2nd level labels are organized in 7 general categories, for example "Computer Science - Data Structures and Algorithms". We have simply used the 2nd level labels, without taking into consideration the hierarchical structure of the taxonomy. In total, around 450 K abstracts have been used.
- BASE (www.base-search.net). This is an open access archive of scientific documents operated by Bielefeld University Library. The adopted categorization method is DDC (http://dewey.info), which is rather old but it covers a wide range of scientific material. We have adopted the first 2 DDC levels, i.e. 100 content classes were used in total. The BASE dataset contains manually annotated data in several languages, however since the current classifier version functions on English texts, we have used almost 35 K annotated documents in the English language.
- WoS (Web of Science, http://thomsonreuters.com/web-of-science). This is a large citation index categorized in approximately 180 class labels (non-hierarchical). We have used about 18 K labelled abstracts.

These datasets have been used to train the classification and visualization modules. Furthermore, arXiv was used by the funding module to detect the fundings of all related publications. *Only the abstracts* have been used for training and testing our methods.

5 Results

5.1 Content Classification Performance

In order to evaluate the classification submodule, we have used the arXiv dataset (the largest among all three datasets). It has to be noted the arXiv dataset is a multi-label one: this means that each document can belong to more than one class labels at the same time. However, it has been found that in 95 % of the cases the arXiv abstracts are annotated to a single label. Therefore, the classification task is quite hard for that case (only most dominant classes are counted as correct). However, we have computed the recall rate for a number of returned results by the classification module. Also, we have used several thresholds of the probabilistic ration described above. These results are presented in Fig. 1. Note that, since most of the arXiv abstracts are single-label, there is no meaning computing the precision rate for this dataset.

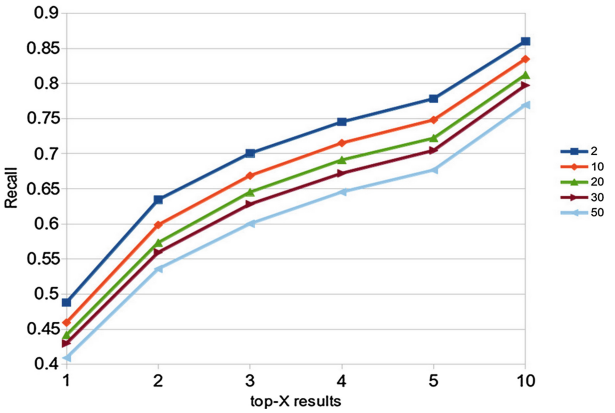


Fig. 1. Classification performance for different probability ratio thresholds. The arXiv dataset has been used.

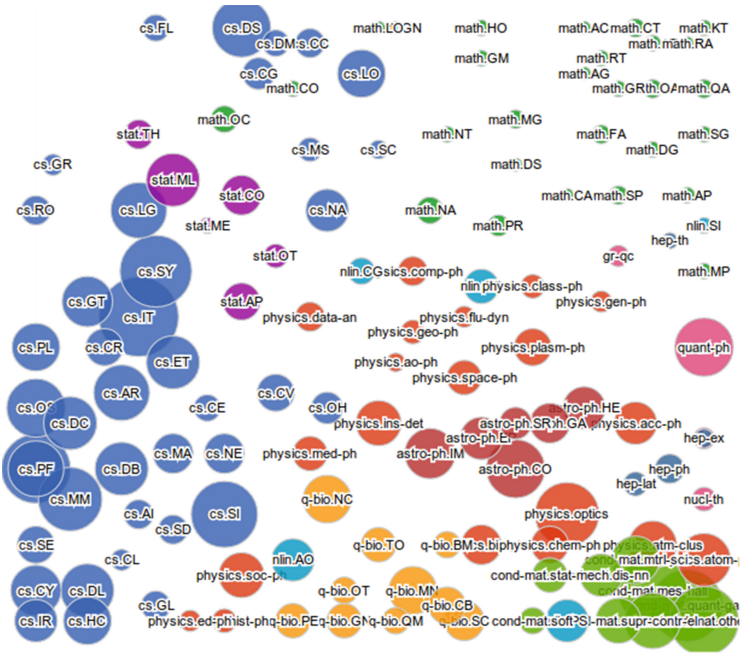
A first obvious observation of the classification performance results is that if larger dictionaries are used (lower threshold in the dictionary compilation procedure) the performance is boosted. However, the more dictionary terms used, the more the computational time cost. To make this clearer, in Table 1, we present the average classification times per abstract (in milliseconds) for different dictionary sizes (i.e., different thresholds adopted in the dictionary training procedure). Two general conclusions can be drawn from the Table: first, the average execution time per abstract is not increased when the process is batch-executed for a very large number of abstracts (actually it decreases - 10 % to 40 % in some cases); second, there is a huge drop (50 %) in the average execution time per abstract for $T = 10$ (related to the $T = 2$ case).

Table 1. Average execution times (msecs) per abstract of the classification module. Higher dictionary thresholds (less dictionary terms) lead to faster classifications.

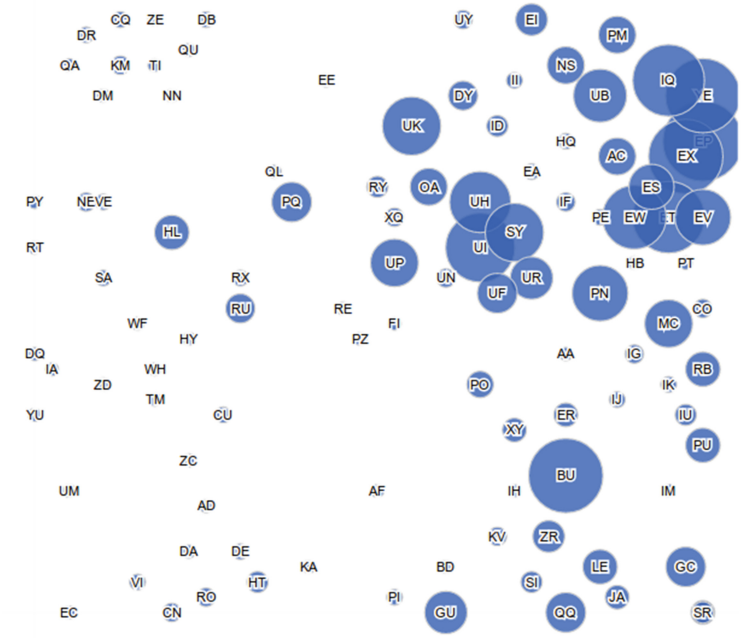
	Threshold				
#abstracts	2	10	20	30	50
10	63	27	21	20	17
15000	57	23	15	13	10

5.2 Funding Mining Performance

The funding mining submodule has been extensively evaluated against a large number of publication corpora from library resources, some of which are: ArXiv, PLoS, PUMA, OA set from Europe PubMed Central, etc. The resulting accuracy was over 99 %, with almost all mistakes or false alarms given with very low confidence values. Therefore, they can easily be identified and removed during a

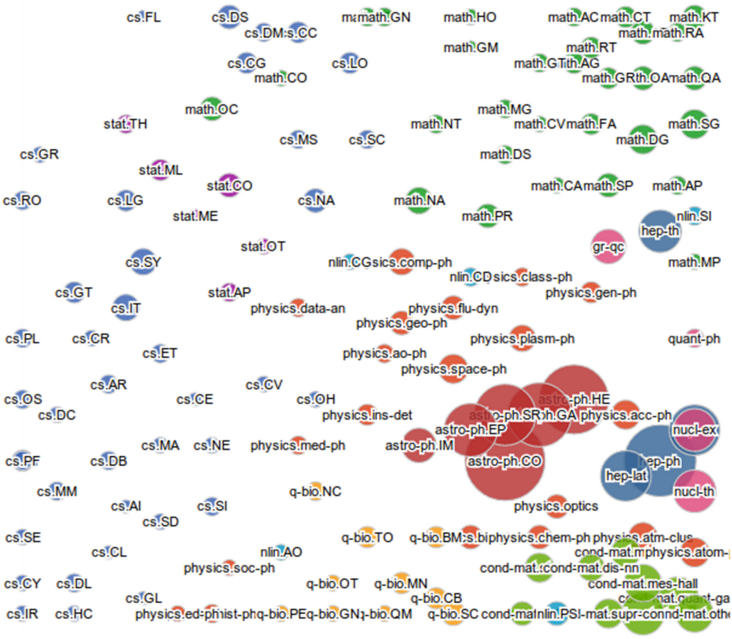


(a) (a) Trained on ARXIV

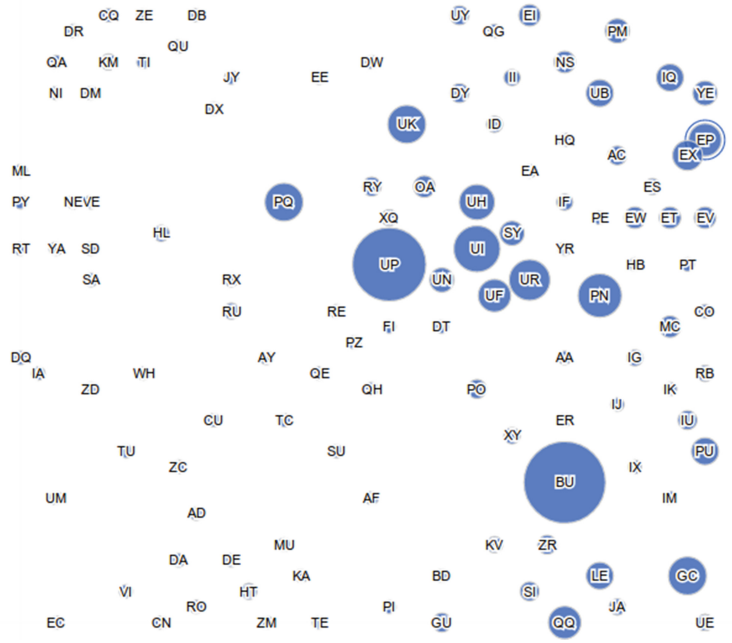


(b) (b) Trained on WOS

Fig. 2. Content distribution of the **ICT**-funded (FP7) publications from the arXiv dataset, using classification and visualization models that have been trained on two taxonomies: ARXIV and WOS



(a) (a) Trained on ARXIV



(b) (b) Trained on WOS

Fig. 3. Content distribution of the **PEOPLE**-funded (FP7) publications from the arXiv dataset, using classification and visualization models that have been trained on two taxonomies: ARXIV and WOS

curation phase. The estimation of this performance has been found by verifying each funding match/results given by the algorithm by manual curation. However, we cannot give precision and recall figures because the datasets are not labelled. We are only able to check the matches returned by the text mining. Furthermore, as we continue to process more and more publications, the module's filtering rules can be further fine-tuned, based on the feedback received after curation. As far as the time complexity is concerned, the module's processing times are very short: between 2180 and 5090 full text articles per minute depending on the dataset (about 3600 full text publications/min on average).

5.3 Content Visualization Examples

Figures 2 and 3 present the content visualizations for two different funding categories, namely the FP7-ICT and FP7-PEOPLE calls. Both corpora have been built using the funding mining submodule, applied on the whole set of the ARXIV dataset. The result was more than 4000 fund-related publications for both funding cases. These two corpora of publications have been automatically classified and visualized using the proposed method (for classification threshold

Table 2. Some of the category labels and names from the arXiv and WOS taxonomies

arXiv		WOS	
astro-ph.GA	Astroph. - Galaxy Astrophysics	BU	Astronomy - Astrophysics
astro-ph.SR	Astroph. - Solar and Stellar	GC	Geochemistry and Geophysics
cs.DB	CS - Databases		
cs.DL	CS - Digital Libraries	GU	Ecology
cs.IT	CS - Information Theory	IQ	Engineering, Electrical and Electronic
cs.LG	CS - Machine Learning		
cs.PF	CS - Performance	EX	Computer Science, Theory and Methods
cond-mat.other	Condensed Matter - Other	HL	Health Care Sciences
hep-lat	High Energy Physics - Lattice	LE	Geosciences, Multidisc.
hep-th	High Energy Physics - Theory	PU	Mechanics
physics.geo-ph	Physics - Geophysics	RU	Neurosciences
physics.optics	Physics - Optics	UB	Physics, Applied
physics.space-ph	Physics - Space Physics	UK	Physics, Condensed Matter
quant-ph	Quantum Physics	UI	Physics, Multidisc.
q-bio.CB	Quantitative Biology - Cell Behavior	UP	Physics, Particles and Fields
stat.ML	Statistics - Machine Learning	YE	Tellecom.

$T = 20$). In each case, two classification-visualization models have been adopted: one trained on the ARXIV dataset and another on the WOS dataset. Therefore, for each of the two funding categories we show two visualizations. Note that the meaning of the colors for the arXiv model are not important; they just indicate the first-level (general) classes of the arXiv taxonomy. In the case of the WOS visualization model a single color is presented, since that taxonomy is composed of a single level of classes. Finally, Table 2 presents a list of classes (for both taxonomies), in order to make the interpretation of the figures more clear. This is not the whole list of classes but the most representative in the visualization results[6].

6 Conclusions

We have presented a system towards visualization of corpora of scientific publications using supervised knowledge. Special focus has been given on implementing the whole system in the context of a data processing workflow, through the utilization of an extensible relational database. In order to achieve direct integration of both the classification and the visualization modules in the data processing workflow, we have selected to implement simple but effective approaches (e.g. the Naive Bayes classifier). Performance results have been given on the individual submodules, and a series of visualization results has been presented for specific funding schemata. Our future research efforts will focus to the following directions: (a) evaluation of the content visualization results (b) embedding more semantically important information in the results (e.g. text tags) (c) model temporal evolution of the extracted visualizations to discover trends.

Acknowledgments. The research leading to these results has received funding from the EU's FP7 under grant agreement no. RI-283595 (OpenAIREplus).

References

1. arXiv: (https://www.arxiv.org) Cornel University Library article archive
2. Bird, S.: NLTK: the natural language Toolkit. In: COLING/ACL on Interactive Presentation Sessions, pp. 69–72. Association for Computational Linguistics (2006)
3. Rish, I.: An empirical study of the Naive bayes classifier. In: IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, vol. 3, pp. 41–46 (2001)
4. Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 4th edn. Academic Press Inc., New York (2008)
5. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer-Verlag New York Inc., New York (2001)
6. Madis: https://code.google.com/p/madis/Complex data analysis/processing made easy

[6] More visualization examples are available in http://www.di.uoa.gr/~tyiannak/PubVisulatisations/contentAnalysis.html.