Technical documents classification

S. Chagheri, Catherine Roussey, Sylvie Calabretto, C. Dumoulin

in : 13th International Symposium on the Management of Industrial and Corporate Knowledge, 8-9 juin 2011

Technical Documents Classification

Samaneh Chagheri, Catherine Roussey, Sylvie Calabretto
Université de LYON, CNRS, LIRIS UMR 5205
Lyon, France
samaneh.chagheri,sylvie.calabretto@insa-lyon.fr
catherine.roussey@liris.cnrs.fr

Cyril Dumoulin
CONTINEW
Roanne, France
cyril.dumoulin@continew.fr

***Abstract*—This research takes place in an industrial context: the CONTINEW Company. This company ensures the storage and security of critical data and technical documentation. The term "technical documentation" refers to different documents with product-related data and information that are used and stored for different purposes, such as user manuals and product specifications. They are strongly structured, but different authors have used different styles and models for document construction. The management of this increasing volume of documents requires document classification in order to retrieve information quickly and to construct a standard model for each category of documents.**

Structural document; document classification; support vector machine; vector space model

I. Introduction

This research takes place in an industrial context: the CONTINEW Company. This company ensures the storage and security of critical data and technical documentation. The term "technical documentation" refers to different documents with product-related data and information that are used and stored for different purposes, like user manuals, product specifications, manufacturing documents and product presentations. Each purpose indeed present a category. Documentation is necessary in the life cycle of an innovative product, from its functional design to marketing. Indeed without such documents, the products can neither be manufactured nor used according to their complexity. Readers of technical documentation may be the end user of the product or the product administrator or technician. It describes specification and requirements for the product to function as designed and it contains more elements like tables and schema. In a general collection the documents are more textual and are strongly structured. Structure here means logical structure like title, chapters, sections, figures, paragraphs, etc. These documents are created by different authors using different styles and forms to present them. A classification system is needed for manipulating such a voluminous collection in order to facilitate information retrieval and to construct a standard model for each category. Traditional classification systems do not consider the structure of a document. They mostly focus on representing the document content as a bag of words without considering the structural elements in which the words appear. But taking into account the structure facilitates document presentation and improves the classification precision. These systems exploit the available structural information in documents, as marked up in XML, in order to implement a more accurate classification strategy and return document structural elements instead of complete documents. However, much of the information is contained in the text fields not just in tag labels, therefore, applying a method which exploits the structure of document and its content is promising.

In this article, we have proposed a representation method which is an extension of the vector model of Salton [8] adjusting the calculation of the tf*idf by considering the structural element instead of the entire document. Document is represented as a tree in which nodes are structure, and leaves are document text. The rest of the article is organized as follows. Section 2 presents the different approaches proposed in the literature to XML document classification. Section 3 describes the basic methods for classification. Section 4 presents our proposal on document representation and feature construction. Section 5 describes the experiments and results, and section 6 presents the conclusion and further works.

II. Related Works

The continuous growth in XML documents has caused different efforts in developing classification systems based on document structure. Document representation has to be done before the classification process. The representation models can be divided into three groups: The first groups are the models which do not consider the structure of document. These works focus on representing the document content as a bag of words to classify them. They are the most studied classification methods. The second group are the models which take into account only the structure of a document in order to classify them without considering the document content. For example, Wisniewski [10] uses Bayesien model to generate the possible DTDs of a documents collection. A class represents a DTD. They are interested only on document structure for classification. Reference [1] and [3] have also classified the documents by only document structure trees similarities. Finally, the third group is composed of the models which consider both structure and content of XML documents in representation.

Mostly the classification systems use vector space model for document representation, the difference is based on selecting vector features and their weight computation. In [4] each vector feature can be a word or a tag of XML document. The *tf*ief (Term Frequency * Inverse Element Frequency)* is used for calculating the weight of words or tags in documents.

In [9] a feature can be a path of the XML tree or a path following by a text leaf. The term weight is based on *tf*idf*. This allows taking into account either the structure itself or the structure and content of these documents. Reference [14] presents also vector model containing terms or XML tree paths as vector elements. Ghosh [5] has proposed a composite kernel for fusion of content and structure information. The paths from root to leaves are used as indexing elements in structure kernel weighted by *tf*idf*. The content and structure similarities are measured independently. A linear combination of these kernels is used finally for a content-structure classification by SVM. Wu [11] proposes a bottom up approach in which the structural elements are document tree nodes and the leaves are textual section of each element. First the terms in leaf nodes are identified, and their occurrences are extracted and normalized. Then the key terms are substantiated with the structural information included in the tags by the notion of key path. A key path ends at a leaf node that contains at least one key term for a class. By using the key terms set and key path the similarity is computed between new documents and class models. Yan [12] proposes a document representation by vectors of weighted words, a vector for each structural element in a document. In this method a weight is associated to each structural element according to its level in the document tree. Then the weight of words is calculated based on its frequency inside the element and the importance of this element. Yang in [13] and [14] has proposed an extension of the vector model called the Structured Link Vector Model (SLVM). In his model a document is represented by a set of term vectors, a vector for each structural element. The term weight is calculated by term frequency in each document element and the inverse document frequency of term.

The model that we propose belongs to the third group, constructing the document feature vector by structure and content. In which each vector feature is a couple of word and structure element. The weight of the word is calculated not only on its frequency in document and collection, but on its position in document. In our calculation we assign a value to each structural element according to its level in the XML document tree.

III. Document Classification

Classification can be divided in two principal phases. The first phase is document representation, and the second phase is classification. The standard document representation used in text classification is the vector space model. The difference of classification systems is in document representation models. The more relevant the representation is, the more relevant the classification will be. The second phase includes learning from training corpus, making a model for classes and classifying the new documents according to the model. In this section we are going to explain the vector space model followed by a presentation of the SVM classification algorithm which is used in our approach.

A. Document Representation by VSM

The Vector Space Model (VSM) proposed by Salton (Salton, 1968) is a common technique for document representation in classification. In this model each document is represented as a vector of features. Each feature is associated with a weight. Usually these features are simple words. The feature weight can be simply a Boolean indicating the presence or absence of the word in document, its occurrence number in document or it can be calculated by a formula like the well known *tf*idf* method.

So, the feature vector of document d of collection D with n distinct terms is represented as follows:

$$d_d = [w_{1,d}, w_{2,d}, .., w_{n,d}] . \quad (1)$$

$$w_{i,d} = tf_{i,d} * idf_i \quad (2)$$

$w_{i,d}$ is the weight of term i in the document d, where tf is the frequency of term i in document d and $idf_i = \log(|D|/|D_i|)$ is inverse document frequency, $|D|$ is the total number of the documents in collection D, and $|D_i|$ is the number of documents in collection containing the term i.

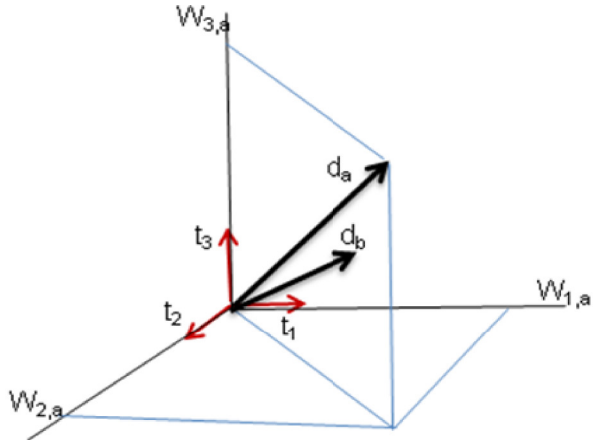

Figure 1. A document feature vector: a & b represent two documents of the collection.

But in classical vector space model, document structure is not considered. For taking into account the notion of structure in document representation an extension of VSM seems promising.

B. SVM Classifier

The Support Vector Machine (SVM) proposed by Vapnik [2], is a supervised learning algorithm that can be applied to classification. It is a binary linear classifier which separates the positives and negatives examples in a training set. The method looks for the hyperplane that separates positive examples from negative examples, ensuring that the margin between the nearest positives and negatives is maximal. The effectiveness of SVM is superior to other methods of text classification. SVM makes a model representing the training examples as the points in a dimensional space separated by the hyperplane, and it uses this model to predict a new example belongs to which side of this hyperplane. The examples used in searching the

hyperplane are no longer used and only these support vectors are used to classify a new case. This makes a very fast method.

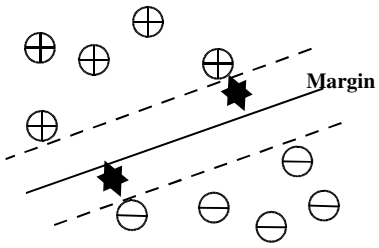

Figure 2. Maximum-margin hyperplane

IV. General Overview of our Approach

In this article we propose inputsa combination of content and structure of XML document in the vector model for document representation. In our vector, each feature is a couple of (tag: term). Tag corresponds to a structural element in an XML document. The XML document is represented as a tree in which the nodes are the XML tags and the leaves are the textual part in each document element. To construct our features we apply two processes. First, a process is performed on the logical structure document in order to construct the aggregated tree of XML document. Secondly, a lexical process is performed on document content to select the most informative terms. These processes aim to reduce the computational complexity and improve system performance. Then the weight of features is computed. Finally, the document vectors are used as the inputs of the classification system.

A. *Aggregated Document Tree Construction*

We consider the XML document as a tree in which the nodes are structural elements like title, chapter, etc. The arcs of this tree represent the inclusion relation between nodes, and the leaf nodes contain the document text. The depths of nodes in document tree are important. Thus we consider that two nodes with the same label localized at different depths are two different structural elements. The structural element represents a node type.

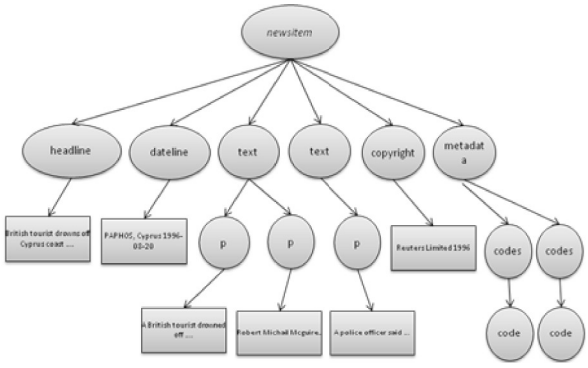

Figure 3. XML document tree

We modify document tree organization by aggregating the nodes with the same label and localized at the same depth. For example, all paragraphs in a section are aggregated to a single node “paragraph” which contains all terms of these paragraphs. We take into account only the leaf nodes which contain text. We assume that all documents of collection are homogeneous. So, a label becomes an adequate identifier of the structural element. Also, we filter some tags which are not representative for document semantic based on a tag list made manually.

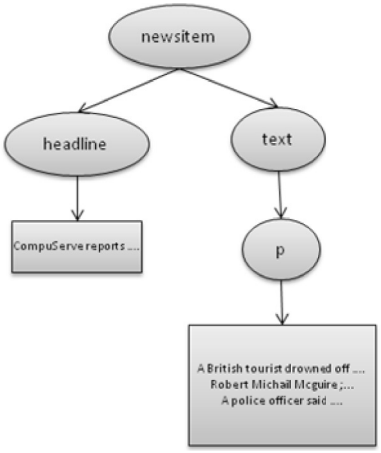

Figure 4. XML document aggregated tree

B. *Feature Extraction*

After constructing the aggregated tree, we extract the terms in each node. We apply a series of linguistic analysis. First we extract the terms lemma and their part of speech using TreeTagger. Secondly we filter the words, just nouns and verbs are kept and the other words are removed. Then the words are replaced by their lemma. The results of such analysis are called term. Therefore, each feature is constructed by combining the node label and the term, for example (title: technical) or (p: cleaning). Such feature represents the document structure and content. Our hypothesis is that a term appearing in two different structural elements should be considered as two different features with different weights. Indeed, moving down in document tree, from root to leaf, decreases the elements importance. For example the word “classification” appearing in the document title and in a paragraph of a document composes two different features: (title: classification) and (paragraph: classification). And the weight of the first feature is more important than the second.

C. *Weight Computation*

We assume that terms occurring in different structural elements have different importance. For calculating the weight of features we use an extension of traditional tf*idf on structural element level instead of document level. We alsot take into account the importance of the structural element in which the term has appeared. We assume that the deeper a node is in the document tree, the less it will be important. Therefore, the weight of feature will be calculated as below:

$$W_{i,e,d} = TF_{i,e,d} * IDEF_{d,e} * IED_e \quad (3)$$

$$IDEF_{d,e} = log \frac{|D_e|}{|D_{e,i}|} \quad (4)$$

$$IED_e = log \frac{L_d + 1}{l_{d,e}} \quad (5)$$

Where i, e, and d represent respectively term i, node label e, like "paragraph" and document d in the collection.

The Term Frequency $TF_{i,e,d}$ is the number of occurrences of the term i in structural element type e inside the document d. $IDEF_{d,e}$ is the Inverse Document Element Frequency with $|D_e|$ as the number of documents in the collection having a node label e, and $|D_{e,i}|$ as the number of documents having node e containing the term i. IED_e is the Inverse Element Depth that represents the importance of the node e in the collection. Where L_d is the depth of document tree, and $l_{d,e}$ is the depth of the node e in this document.

After extracting all features in documents and calculating their weight, document vector is constructed and SVM is used for learning and classifying.

V. Experiments and Results

For experimentation we have used SVM^{light} [7] for learning and classifying the documents. It is an implementation of the SVM algorithm. This algorithm has scalable memory requirements and can handle problems with many thousands of support vectors efficiently.

A. Test Collection

Experiments were performed on the Reuters Corpus Volume 1 (RCV1) which includes over 800,000 English language news stories. Reuters is a leading global provider of financial information, news and technology to financial institutions, the media, businesses and individuals. The stories in this collection are formatted using a consistent XML schema. Reuters collection has been used by many researchers in the field of information retrieval and machine learning. All the stories in RCV1 have been coded for topic, region (geography) and industry sector and they are multiclass.

B. Collection Pre-processing

In order to perform a mono classification we have used the first topic code in documents as the class of documents. Also the unclassified documents are removed from collection. We have used 800 documents of collection by considering 400 positive and 400 negative examples for a topic class called "GCAT". After analyzing the structural elements in documents, we have selected the header and paragraphs tags in stories. These tags seem more representative for document topic. Other tags are removed from document tree. TreeTagger and Tokenizer of platform GATE has been used to extract the terms.

C. Experimentation

We performed two experiments on chosen collection by using SVM. The first one called "content and structure" is an implementation of our proposed method in which the document vector is constructed by tag and term, and where the features weight is also calculated using our proposed formula. The second one called "content only" uses a vector of simple terms weighted by standard tf*idf. The application has been implemented in java except SVM^{light} algorithm which is on C. K-fold Cross validation has been used as the method for evaluating the system performance.

Cross-validation is one of the several approaches to estimate how well the model we've just learned from some training data is going to perform on new arriving data. We have applied 4 folds in cross validation, each fold including 200 documents (positives and negatives examples). The results of the classifications are shown in table I. Precision and recall as the most widely used metrics for evaluating the correctness of classification have been used. Another used metric is F-Measure that combines precision and recall. The results demonstrate that including structure improves the classification accuracy.

TABLE I. Classification Results

Vector features	Results		
	Precision	***Recall***	***F-Measure***
Content & structure	0.96	0.92	0.94
Content only	0.93	0.85	0.89

VI. Conclusion and Future Works

In this article we have proposed a model for XML document classification. We proposed a combination of structure and content in document representation by the vector model. Our features are couples of (tag: term), all weighted by an extension of tf*idf taking into account the terms position in the documents tree. Document tree is achieved by filtering non relevant structural elements and aggregating the nodes with same label and same path in tree. Classification is performed in order to make a model for each category of technical documents. This model is used to evaluate the technical documents quality and adapting them with standard models.

We have done our experimentation on Reuters XML news collection which is a particular collection with homogenous documents. In this collection all documents share same logical structure. In future works we are going to improve our proposition in order to execute the test on CONTINEW corpus which contains heterogeneous documents created by different industries for different categories, and generating document model.

References

[1] Aïtelhadj, A., Mezghiche, M., & Souam, F. (2009). Classification de Structures Arborescentes: Cas de Documents XML. *CORIA 2009* , 301-317.

[2] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 273-297.

[3] Dalamagas, T., Cheng, T., Winkel, K.-J., & Sellis, T. (2005). Clustering XML Documents Using Structural Summaries. *EDBT* , 547-556.

[4] Doucet, A., & Ahonen-Myka, H. (2002). Naive Clustering of a Large XML Document Collection. *INEX Workshop 2002* , 81-87.

[5] Ghosh, S., & Mitra, P. (2008). Combining Content and Structure Similarity for XML Document. *ICPR*, 1-4.

[6] Joachims, T. (1999). Making large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press*, 169-184.

[7] Joachims, T. (1999). Making large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press*, 169-184.

[8] Salton, G. (1968). Search and retrieval experiments in real-time information retrieval. (C. University, Ed.) 1082-1093.

[9] Vercoustre, A.-M., Fegas, M., Lechevallier, Y., & Despeyroux, T. (2006). Classification de documents XML à partir d'une représentation linéaire des arbres de ces documents. *EGC 2006* .

[10] Wisniewski, G., Denoyer, L., & Gallinari, P. (2005). Classification automatique de documents structurés. Application au corpus d'arbres étiquetés de type XML. *CORIA 2005 Grenoble*, 52-66.

[11] Wu, J., & Tang, J. (2008). A bottom-up approach for XML documents classification. (ACM, Ed.) *ACM International Conference Proceeding Series; Vol. 299* , 131-137.

[12] Yan, H., Jin, D., Li, L., Liu, B., & Hao, Y. (2008). Feature Matrix Extraction and Classification of XML Pages. *APWeb 2008 Workshops* , 210-219.

[13] Yang, J., & Wang, S. (2010). Extended VSM for XML Document Classification Using Frequent Subtrees. *INEX 2009*, 441-448.

[14] Yang, J., & Zhang, F. (2008). XML Document Classification Using Extended VSM. *INEX 2007*, 234–244.

[15] Yi, J., & Sundaresan, N. (2000). A classifier for semi-structured documents. *KDD '00,* 340-344.