Document Classification Utilising Ontologies and Relations between Documents

Katariina Nyberg[*] Tapani Raiko[†] Teemu Tiinanen[†]
Eero Hyvönen[*]

ABSTRACT

Two major types of relational information can be utilized in automatic document classification as background information: relations between terms, such as ontologies, and relations between documents, such as web links or citations in articles. We introduce a model where a traditional bag-of-words type classifier is gradually extended to utilize both of these information types. The experiments with data from the Finnish National Archive show that classification accuracy improves from 70% to 74% when the General Finnish Ontology YSO is used as background information, without using relations between documents.

Keywords

document classification, ontologies, relational models

1. INTRODUCTION

More and more documents are produced in the modern information society, and stored in digital form in archives. This creates the need for developing convenient ways of classifying documents automatically for Information Retrieval retrieval (IR) [24, 23].

This paper investigates learning techniques for automatic document classification. The idea is to extend traditional logistic discrimination learning [3] by combining it with relational background knowledge based on ontologies [31, 14]. As a case study, we explore the ways in which 7252 categorised digital documents of the National Finnish Archive, described using the SÄHKE metadata model [16], could be classified automatically. This paper shows that learning of classes for documents improves when knowledge about the meaning and relationships of the words, based on ontological information, is added to the system.

In the following, we first shortly introduce IR models, dimensionality reduction, and ontologies used as a basis for this paper. After this, the proposed method and data preparation are explained, and the experimental results are presented. In conclusion, related work is discussed.

2. BACKGROUND

2.1 Models for Information Retrieval

In the traditional bag-of-words model of IR, a document is represented as an unordered collection of terms that occur in the document. This model is based on the assumption that two documents with similar bag of words representations are similar in content [23, p. 107]. The bag of words model is a simplified representation of a document, because it assumes that the document's terms are independent of each other [24, p. 237], that they are all of equal importance and that the term's ordering is of no importance [23, pp. 105].

The number of times a term occurs in a document, or across the document set, doesn't necessarily provide information about the contents of the document. However, frequences are important in ranking the documents in the Vector Space Model (VSM) [23]. Here a document is represented by a vector of the weights for all of its terms. The terms that do not occur in the document have the weight 0. The relevance of a document to query, represented also as a vector of terms, can be defined as similarity measure (e.g. the cosine) between the query and document vectors.

A widely used model for term weigthing is the tf-idf method, where the term weight is increased based on its frequence in a document and decreased based on its frequence arcoss the document set. To measure these effects, the term frequency in a document is defined as

$$tf_{t,d} = \frac{\text{number of occurrence of the term } t}{\text{number of terms in the document } d} \text{ [3, p. 64].} \quad (1)$$

and the inverse document frequency as

$$idf_t = \log \frac{N}{df_t}, \quad (2)$$

where N is the total number of documents, and df_t is the document frequency, i.e. the number of documents in which the term t occurs [23, p. 108]. By multiplying the term frequency with the inverse document frequency, the tf-idf weight is defined for a term t in a document d:

$$tf\text{-}idf_{t,d} = tf_{t,d} \times idf_t, \quad (3)$$

This model will be a basis of representing documents in our learning method.

[*] Department of Mediatechnology, Aalto University School of Science and Technology, firstname.lastname@tkk.fi

[†] Adaptive Informatics Research Center, Department of Information and Computer Science, Aalto University School of Science and Technology, firstname.lastname@tkk.fi

MLG '10 Washington, DC USA
Copyright 2010 ACM 978-1-4503-0214-2 ...$10.00.

2.2 Dimensionality Reduction

The dimensions of vectors if VSM are typically large, which raises the question, whether dimension reduction techniques could be applied here for better retrieval performance. A candidate for this is Principle Component Analysis (PCA) that projects the variables on to a lower dimension in such a way that the variance of the data points is maximized [3, pp. 108]. This paper will utilize PCA for dimension reduction.

PCA starts by choosing the first principal component, which is an eigenvector of the covariance matrix of the data. The eigenvector with the largest eigenvalue has the largest variance, thus the first principal component is the eigenvector of the covariance matrix with the largest eigenvalue [7, p. 562]. Each next principal component is an eigenvector of the covariance matrix with the next largest eigenvalue and thus all components are orthogonal to each other [3, p. 110].

PCA can be solved by spectral decomposition of the estimated covariance matrix of the data matrix $\mathbf{X}$:

$$\frac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{T}} = \mathbf{C}\mathbf{D}\mathbf{C}^{\mathrm{T}}, \tag{4}$$

where N is the number of examples in the data (the number of columns in $\mathbf{X}$), $\mathbf{D}$ is a diagonal matrix containing the eigenvalues and $\mathbf{C}$ contains the appropriate eigenvectors. The data is assumed to be centralised by substracting the mean of each row from it. For the dimension reduction of $\bar{x}$ of size $N \times 1$ to $\bar{z}$ of size $M \times 1$, where $M < N$, the following must apply:

$$\bar{z} = \mathbf{U}^{\mathrm{T}}\bar{x}, \tag{5}$$

where $\mathbf{U}$ contains the M eigenvectors of $\mathbf{C}$ corresponding to the greatest eigenvalues [3, pp. 108].

Application of PCA to document analysis is often called Latent Semantic Indexing (LSI) [15]. LSI has the ability to correlate semantically related terms in a collection of text by mapping them close together in the low-dimensional representation. In that sense, LSI and the use of term ontologies have the same goal. Note that one can use unclassified documents for the dimensionality reduction. The work on LSI has continued for instance towards kernel methods [30, 13, 5, 8]. In the current paper, we will combine LSI ideas with manually constructed term ontologies.

2.3 Ontologies

Ontology in philosophy means the theory of being, existence and reality. The field of Artificial Intelligence (AI) has taken the word ontology to mean something different though related. Studer et al. [31] combine the ontology definitions of [18] and [9] and define an ontology as "a formal, explicit specification of a shared conceptualisation". When people communicate about the world, the words they use hold a meaning for them, but for the machines words are just a meaningless symbols unless formally defined. Therefore in AI there is a need for putting the words into a structure of concepts with well-defined semantics, i.e. an ontology. Formal description of the concepts allow for an ontology to be machine interpretable [31, p. 25].

The concepts in an ontology are related to human readable words (literals) and to each other through semantic relations. Semantic major relations in ontologies and vocabularies include [1, 26]:

- **Hyponymy** The apple is a subconcept of the fruit, where the fruit is the hypernym of the apple and the apple is a hyponym of the fruit. Hyponymy, a hierarchical subclass of relation exists, when all the instances of a concept X, that is a subclass of a concept Y, are also instances of Y [18, p. 28]. For example, the Titanic is an instance of "watercraft", which is a subclass of the concept "vehicle". Thus all instances of the class watercraft are also instances of the class vehicle.
- **Meronymy** A hierarchical relation different from the hyponymy is the part-of relation. A "branch" is not a subclass of a "tree", but part of it. The distinction is important to make, because in a hyponymy concepts inherit the characteristics of their broader concepts [19], but not in a meronymy.
- **Associative** An ontology can also contain semantic relations that are not hierarchical. The different ways in which terms can be associated with each other proves to be quite challenging to be modeled, because there are so many different reasons for associating a term to another. For example, "rain" is associated with an "umbrella", because rain is the reason for using an umbrella. "Water" can be associated with a "well", because that is where water could be carried from.

Words do not necessarily diambiguate uniquely meanings, due to synonymy and polysemy/homonymy (e.g. crane as a bird species vs. a construction rig). In ontologies meanings are distinguished from each other by unique concepts identifiers, but ambiquities remain in mapping literals words with concetps.

Attaching metadata to a document is called annotation; in ontology-based annotation metadata is connected to ontologies [20]. Ciravegna et al. [11] note that document annotation requires a lot of manual work and argue for the need of information extraction to make the process automatic or at least semi-automatic.

Ontologies are widely used on the semantic web, and W3C standards[1] are often used in practise for representing them: Resource Description Framework (RDF), RDF Schema [10], and Web Ontology Language (OWL) [6]. Here concepts of ontologies are resources that are identified with a unique Uniform Resource Identifier (URI) and are described with properties, which themselves are resources with URIs, too. Everything is described using triples of form

$< subject, predicate, object >$,

where the subject is the resource to be described, the predicate is its property, and the object the value of the property. The object can be either a resource or a literal piece of data. [25] RDF triples constitute labelled directed graphs, and this data model will be used in our case study.

2.4 General Finnish Ontology YSO

The ontology to be used in this paper is the General Finnish Ontology YSO[2] of some 23,000 general concepts. It was developed by restructuring the commonly used Finnish General Thesaurus YSA into an ontology [22]. YSA is based on the standard thesaurus relations [1] narrower term (NT) (e.g. fruit NT apple), broader term (BT) (e.g. Helsinki BT Finland), and related term (RT) (e.g., umbrella RT rain) between the thesaurus' terms. The NT/BT relations were analysed and changed into hyponymy and meronymy relations, and the hyponymy hierarchies were completed by restructuring and introducing addional concepts. In addition, the associative RT relations were checked in the new structure.

YSO's concepts are labelled in Finnish labels from YSA and with their equivalent Swedish terms from the General Swedish Thesaurus Allärs. YSO contains also English labels.

3. PROPOSED METHOD

Throughout the presentation, we will be using the following notation:

[1] http://www.w3.org/standards/semanticweb/

[2] http://www.seco.tkk.fi/ontologies/yso/

symbol	range	stands for
d	1...N	documents
t	1...T	terms
i	1...K	classes

Each document is represented by a vector $\bar{x}_d \in \mathbb{R}^T$, which contains the tf-idf weights $tfidf_t \in \mathbb{R}$ for each term in the document and can be written as $\bar{x}_d = [tfidf_1 \; tfidf_2 \; \ldots \; tfidf_T]^{\mathrm{T}}$. As noted in Section 2.1 some of the elements might be 0, because the respective term does not occur in that document. The matrix $\mathbf{X}$ contains all $\bar{x}_d$, $\mathbf{X} = [\bar{x}_1 \; \bar{x}_2 \; \ldots \; \bar{x}_N]$, where $\mathbf{X} \in \mathbb{R}^T \times \mathbb{R}^N$.

3.1 Logistic Discrimination

We start from traditional logistic discrimination (see e.g. [3]) for learning to predict the class c_d for each document. The classifier has as parameters a vector of the weights of each term for the linear discriminant of the class i. The vector is

$$\bar{w}_i = [w_1 \; w_2 \; \ldots \; w_T \; w_{T+1}], w_t \in \mathbb{R}.$$

We also define the matrix $\mathbf{W} \in \mathbb{R}^K \times \mathbb{R}^{T+1}$, which contains the vector $\bar{w}_i$ for each class i, $\mathbf{W} = [\bar{w}_1^{\mathrm{T}} \; \bar{w}_2^{\mathrm{T}} \; \ldots \; \bar{w}_K^{\mathrm{T}}]^{\mathrm{T}}$. The basic model[3] for classification probability $P(c_d = i \mid \bar{x}_d, \mathbf{W})$ is

$$P(c_d = i \mid \bar{x}_d, \mathbf{W}) = \frac{\exp\left(\bar{w}_i^{\mathrm{T}} \begin{bmatrix} \bar{x}_d \\ 1 \end{bmatrix}\right)}{\sum_j \exp\left(\bar{w}_j^{\mathrm{T}} \begin{bmatrix} \bar{x}_d \\ 1 \end{bmatrix}\right)}, \forall i, d. \quad (6)$$

We concatenate 1 to each data vector $\bar{x}_d$ so that w_{T+1} takes the role of the bias term, and we do not thus have to include separate bias terms. Each weight w_{it} can be interpreted as the influence of term t for the document to be classified in class i. The weights can be also negative. The denominator in Eq. (6) is for normalising the class probabilities to sum up to one for each document.

As the first modification, the dimensionality of $\bar{x}_d$ is reduced with PCA (see Section 2.2) in the following way:

$$P(c_d = i \mid X_d, \mathbf{W}) = \frac{\exp\left(\bar{w}_i^{\mathrm{T}} \begin{bmatrix} \mathbf{U}^{\mathrm{T}}\bar{x}_d \\ 1 \end{bmatrix}\right)}{\sum_j \exp\left(\bar{w}_j^{\mathrm{T}} \begin{bmatrix} \mathbf{U}^{\mathrm{T}}\bar{x}_d \\ 1 \end{bmatrix}\right)}, \forall i, d. \quad (7)$$

The modified model has fewer parameters since the size of matrix $\mathbf{W}$ drops from $\mathbb{R}^K \times \mathbb{R}^{T+1}$ to $\mathbb{R}^K \times \mathbb{R}^{M+1}$. This is an important step, because the number of terms T is large, which makes the previous model both computationally complex and prone to overfitting.

3.2 Learning

The maximum a posteriori (MAP) estimate for $\mathbf{W}$ is

$$\mathbf{W}_{\mathrm{MAP}} = \arg\max_{\mathbf{W}} p(\mathbf{W} \mid \bar{C}, \mathbf{X}) = \arg\max_{\mathbf{W}} \frac{P(\bar{C} \mid \mathbf{X}, \mathbf{W}) p(\mathbf{W})}{P(\bar{C} \mid \mathbf{X})}, \quad (8)$$

where $\bar{C}$ contains the known classifications for all the documents in the training data $\mathbf{X}$. Because the denominator is the same for each $\mathbf{W}$ and the logarithm of the function will also find the MAP-estimate for $\mathbf{W}$, the estimate can be written as

$$\mathbf{W}_{\mathrm{MAP}} = \arg\max_{\mathbf{W}} \left[\log P(\bar{C} \mid \mathbf{X}, \mathbf{W}) + \log p(\mathbf{W})\right]. \quad (9)$$

We set the priors for each w_{it} in $\mathbf{W}$ independent and Gaussian:

$$p(w_{it}) = N(0, \sigma_w^2) \quad (10)$$

where w_{it} is the weight of the i^{th} class and the t^{th} term. The weight is expected to be around zero and the σ_w^2 is learned from the data using maximum likelihood. [3, p. 262]

The optimisation step in Equation (9) is solved using gradient based optimisation, that converges to a locally optimal solution. Details are omitted here (cf. [3] for more information).

3.3 Enhancing the Analysis with Relations

This section contains the main contribution of the current work. Let us assume that we have as background knowledge, a set of binary $T \times T$ matrices $\mathbf{A}_r$ that contain relations between terms (See Section 4.9 for an example). The knowledge on the terms held by $\bar{x}_d$ is replaced with the following

$$\bar{y}_d = \sum_r^R \alpha_r \mathbf{A}_r \bar{x}_d, \quad (11)$$

where $\alpha_r \in \mathbb{R}$ is (an unknown) weight for each relationship type r. This can be interpreted as augmenting the data by including virtual appearance of related terms in each document. Note that we always include the identity relationship $\mathbf{A}_0 = \mathbf{I}$.

We can note about the dimensionality reduction

$$\mathbf{U}^{\mathrm{T}}\bar{y}_d = \mathbf{U}^{\mathrm{T}} \sum_r^R \alpha_r \mathbf{A}_r \bar{x}_d \quad (12)$$

$$= \sum_r^R \alpha_r \mathbf{U}^{\mathrm{T}} \mathbf{A}_r \bar{x}_d, \quad (13)$$

that vectors $\mathbf{U}^{\mathrm{T}}\mathbf{A}_r\bar{x}_d$ for each r, d can be computed in advance, because they consist of constant factors.

By replacing $\bar{x}_d$ with $\bar{y}_d$ in the previous version of the model equation (7), we get:

$$P(c_d = i \mid \bar{x}_d, \mathbf{W}) = \frac{\exp\left(\bar{w}_i^{\mathrm{T}} \begin{bmatrix} \sum_r^R \alpha_r \mathbf{U}^{\mathrm{T}}\mathbf{A}_r\bar{x}_d \\ 1 \end{bmatrix}\right)}{\sum_j \exp\left(\bar{w}_j^{\mathrm{T}} \begin{bmatrix} \sum_r^R \alpha_r \mathbf{U}^{\mathrm{T}}\mathbf{A}_r\bar{x}_d \\ 1 \end{bmatrix}\right)}, \forall i, d. \quad (14)$$

Further, let us assume that we have as background knowledge also a set of binary $N \times N$ matrices $\mathbf{B}_s$ that contain relations between documents. We would like to further augment the data by including virtual appearance of terms in related documents. We first write the basic model equation (6) in a matrix form that classifies all the documents at once:

$$\bar{c}_i = \frac{\exp\left(\bar{w}_i^{\mathrm{T}} \begin{bmatrix} \mathbf{X} \\ 1\,1\ldots1 \end{bmatrix}\right)}{\sum_j \exp\left(\bar{w}_j^{\mathrm{T}} \begin{bmatrix} \mathbf{X} \\ 1\,1\ldots1 \end{bmatrix}\right)}, \forall i, \quad (15)$$

where the elements of $1 \times N$ vector $\bar{c}_i$ contain the probabilities $P(c_d = i \mid \mathbf{X}, \mathbf{W})$ for each $d = 1, \ldots, N$. Now we can notice that we just need to multiply the data matrix $\mathbf{X}$ from the right in this case.

The final model becomes:

$$\bar{c}_i = \frac{\exp\left(\bar{w}_i^{\mathrm{T}} \begin{bmatrix} \sum_r^R \alpha_r \mathbf{U}^{\mathrm{T}}\mathbf{A}_r\mathbf{X} + \sum_s^S \beta_s \mathbf{U}^{\mathrm{T}}\mathbf{X}\mathbf{B}_s \\ 1\,1\ldots1 \end{bmatrix}\right)}{\sum_j \exp\left(\bar{w}_j^{\mathrm{T}} \begin{bmatrix} \sum_r^R \alpha_r \mathbf{U}^{\mathrm{T}}\mathbf{A}_r\mathbf{X} + \sum_s^S \beta_s \mathbf{U}^{\mathrm{T}}\mathbf{X}\mathbf{B}_s \\ 1\,1\ldots1 \end{bmatrix}\right)}, \forall i. \quad (16)$$

[3] Note that we start building towards the final model equation (16) gradually, because understanding it without the intermediate versions would be difficult.

The model parameters $\mathbf{W}$, $\alpha_r \forall r$, and $\beta_s \forall s$ remain to be trained using gradient ascent. The model is thus learning how to weight different kinds of relations in order to best classify the training data. Initially, the weights α_r and β_s are set to 1 while $\mathbf{W}$ is set to $\mathbf{0}$. R is the number of different kinds of relations between terms (including the identity) and S is the number of different kinds of relations between documents.

Here we provide examples of usefulness of the proposed approach. Let us say we have as background information a matrix $\mathbf{A}_1$ that provides hyponym relations between terms. We would like to classify a document describing apples to the class fruit. We know that the term fruit is strongly related to the class, but it does not appear in the current document. Using the above model, the term apple appears also as term fruit with weight α_1, thus making the classification easier. Also, we know that a particular author often writes about fruit, and we have as background information a matrix $\mathbf{B}_1$ that provides shared-author relations between documents. The terms in the other documents are also included with weight β_1 again making the classification easier.

The computational complexity increase of the proposed method when compared to the basic model in Equation (7) is small assuming that the number of different kinds of relations $R + S$ is small. The number of additional parameters is simply $R + S$ and the time complexity is approximately $R + S$ times larger.

4. DATA PREPARATION

4.1 Metadata model

The Finnish National Archives dictates that any national or municipal organisation that wishes to store digital documents permanently needs to ask for the Archives a permission and has to follow the SÄHKE metadata model. The model is concerned with the digital handling, managing and finally storing of information on official documents concerning national and municipal governments. It dictates the way in which metadata, the information on the official documents, such as the author and the title, is stored. In the SÄHKE metadata model each document is part of a procedure and the metadata of the procedure is also stored. The SÄHKE metadata model forms a standard under which the digital case management system of national or municipal organisations can be formed. [4]

The abstract specifications of the SÄHKE metadata model [16] introduce an archive hierarchy under which actual documents are stored. The archive hierarchy contains the parts of the procedure in which documents are stored. Each document is associated to one or more actions through an XML reference. Each action is then linked to a *case* (cf. Figure 1). Inside a case, a number of *actions* can contain the same document, if the document is of significance to the actions. Each case belongs to one group and each group represents one class of a given classification. Any organisation using the SÄHKE metadata model is an archive creator and holder, and maintains one or more archives. The archive holder is the agent that produces all the information inside the archive [16]. Each archive contains one or more groups under which the cases to be stored are grouped. Groups can also contain sub-groups. Because of this the groups form a hierarchical structure that can also be seen as the classification for the cases, actions and documents.

The archive hierarchy is a direct representation of the hierarchy of the XML-file that holds all the metadata information of one archive holder and its archives. In that XML-file the archive holder is represented as an XML-entity so that the entity contains among others one or multiple archive entities. Each archive entity contains one or several group entities. Each group entity contains one or more cases and may contain also references to its supergroup or subgroups. A case entity contains one or more action entities and action entities contain one or more document entities.

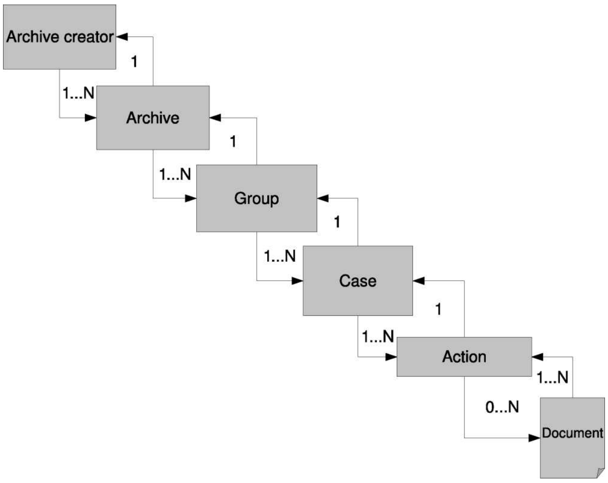


Figure 1: Archive hierarchy of the SÄHKE metadata model as specified in the article on the abstract modelling for the SÄHKE project [16]

4.2 The Archives' Documents

This case study on document classification is based on a data set from the Finnish National Archive (in short the Archive). The semantic Computing Research Group (SeCo) was provided with a test set of the Archive's own case management system. The test data contains documents and a XML-file with all the metadata concerning the information of the archive holder, archive, groups, cases, actions and documents according to the SÄHKE metadata scheme.

The Archive provided the research team with a listing of a classification and its hierarchy. The listing contains a 2-levelled classification with 70 classes and 45 subclasses. Only 13 classes have subclasses and a class with subclasses has on average 3.23 subclasses.

Of the provided classification the test data uses only 67 classes of which 31 are subclasses. In the metadata XML-file these are represented by the group entities. The unused classes of the provided classification are not included in the metadata XML-file.

The set contains 7252 documents that are linked to inquiries directed to the Finnish National Archives. The inquiries are part of the National Archive service. Normal citizens or researchers ask for example for access to certain kind of information that the Archives hold or may hold. The documents are linked to 32325 actions in total. They describe the actions taken during a process, where an inquiry is received and dealt with. Actions describe for example the event when the Archives employee answers to an inquiry. The numerous actions in this data set link to 3469 cases. The cases are categorised under 67 groups. This particular data set is subject to only one archive.

4.3 Transformation of the Metadata to RDF

The metadata of the case management system was read and turned into RDF-form using the Turtle syntax, also denoted by the abbreviation ”TTL”. Each XML element was turned into a resource using the namespace `http://www.narc.fi/onto#` and a local name that consisted of the element's tag name and an arbitrary number identifying the element. A resource class was created from the element's tag name and the class was set as the type of that resource. Every attribute of a element was turned into a triplet with the element resource as the subject, a property created from the namespace and attribute 's local name as the predicate and the attribute's value as a literal object.

4.4 Document Text Extraction

Out of the 7252 documents 2324 were scanned documents and they were image's, mostly of form TIFF. Some of these documents were written by hand and contained even hand written Russian text. Others were written on a machine and therefore an OCR-scan of them would have helped in getting the document text in digital form. Due to lack of resources a sufficient OCR-scan was not performed and these documents were left out of the digital analysis.

In the end 4919 documents could be used for the analysis. A Java programme was written that extracted the text of all documents in various machine readable file formats. The programme read each documents' text into one String variable, sanitised from malicious characters and then passed on to a syntax analyser component.

4.5 Syntactical Analysis of the Text Extracted

For this case study of the documents of the Finnish National Archive the Machinese Syntax[4] component created by Connexor Oy proved to be very sufficient. The component works for all of the Archive's documents' languages, Finnish, Swedish and English. The Machinese Syntax was used on the text extracted from the documents. The component takes as its input text and returns that text in XML-form. It recognises each sentence and numbers them, it numbers each word inside a sentence and turns them into base form and also analyses the syntactical relations between words. [12]

4.6 Parsing the Syntactical Analysis

The analysis from the Connexor Machinese Syntax component in XML-form was parsed using the POKA-tool[5] of the Semantic Computing Research Group. Its class FDGParser written by Olli Alm was originally designed to transform the XML-output of the component to a more efficient XML-form, that the POKA-tool used [2]. For the purpose of this case study, the FDGParser was modified so that it stored the information from the Machinese Syntax analysis into sentence and word objects. This was done in order to store and further process the information from the analysis.

4.7 RDF Transformation

When all document files had been read, sent for analysis, received from analysis and transformed into objects, the objects and the information they held were read into a Jena Model[6]. The RDF was created based on the RDF Schema represented in Figure 2. In this RDF schema the term is in the middle of the focus, because the Machinese Syntax component's analysis provides information for each term rather than for a sentence.

For each term a resource of type http://www.yso.fi/meta/depRDF#term was created for, so that the term's URI consisted of a localname beginning with "term_" and followed by the term's unique ID number. The term resource has three literal properties that store the term's sequence number of the term in the sentence (location), the term's base form (lemma) and its original form (term).

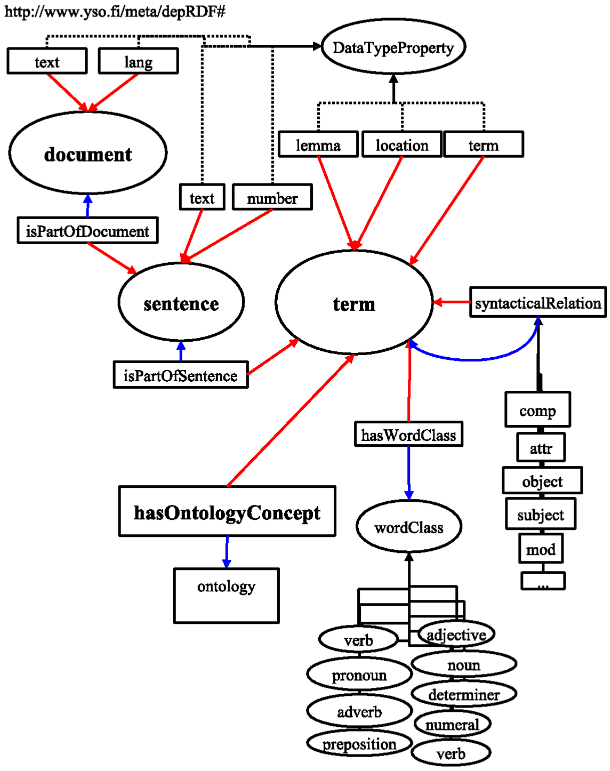


Figure 2: RDF Schema on how information of the documents and their Machinese Syntax Analysis was stored into

The syntactical relation from one term to another is held by various sub-properties of a property called "syntacticalRelation". The sub-properties get their localname from the Machinese Syntax analysis. The relation between a term and its original sentence is marked with the relation "isPartOfSentence".

Each sentence was represented by a resource of type http://www.yso.fi/meta/depRDF#sentence and the resource had two literal properties, the text of the sentence and its sequence number in the document it belonged to. The relation between the document and the sentce was marked with the relation http://www.yso.fi/meta/depRDF#isPartOfDocument. Each sentence resource has a localname "sentence_" followed by its documents unique ID number and its sentence number.

Each document is represented by a resource of type http://www.yso.fi/meta/depRDF#document. A document has two literal properties one of which stores the documents language code (lang) and the other the extracted text of the document. The document resources have localnames that start with "document_" followed by the documents unique ID number.

The word class of each term is recognised as one of nine choices: verb, pronoun, adverb, preposition, adjective, noun, determiner, numeral and conjunction. These were represented in the RDF model by nine resources of type http://www.yso.fi/meta/depRDF#wordClass. The "hasWordClass" property points from a term to one of the nine word class resources.

A term can also be associated with a URI reference to an ontology. In this particular case study the Finnish General Ontology (YSO) was used. It contains terms that have labels at least in

[4] earlier known as The Functional Dependency Grammar (FDG)
[5] http://www.seco.tkk.fi/tools/poka
[6] Jena Model Javadoc: http://jena.sourceforge.net/javadoc/com/hp/hpl/jena/rdf/model/Model.html

Finnish and Swedish, but also some in English[7]. The POKA-tool comprises a method that matches a query with the label in one of the above mentioned languages. The lemmas of each term were given as an input for this method and the result of the method was a list of URIs from the YSO ontology. Most terms were matched with only one URI. Trust was put on the POKA-tool and these matches were not checked. 759 terms, though, were matched with two or more URIs. The multiple alternatives came mostly from polysemous concepts. For example the word child has three different meanings in YSO: a role of a person based on her age, a role of a person belonging to a certain social-economic group and the concept for a family member (its subclasses are daughter and son).

The multiple URIs were checked by hand, the correct YSO reference was selected and other references removed.

4.8 Creating the Data Set

For the machine learning model, the data was transformed into a table with a constant number of columns. Each row contained the appearance (instance) of a term in a document. The class of each instance was the category to which the instance's document belonged to. The category of each document was read from the SÄHKE metadata model. The category of a document is represented by the group node of the metadata schema (see Figure 1). The properties for each individual term was all the information from the Machinese Syntax analysis that was stored into a RDF Schema according to Figure 2.

The parsed result of the Machinese Syntax analysis, the RDF model, was turned into a two-dimensional CSV table. Each row represented an individual term and each column contained the property values for each term. For transforming a RDF model into a two-dimensional table, a programme was written, that created a row for each resource of a certain given class type. In this particular case study this class type was the resource `http://www.yso.fi/meta/depRDF#term`. Each triplet containing the resource of this type as its subject, was taken into account and the information it contained was put into the resulting table followingly: All triplets were iterated and each of their predicates was turned into a column. Then each triplet was gone through again and their object was set as the value of the subjects row and the predicates column. If the object was a resource, its localname was set as the value. If the object was of type Literal[8] then its text was set as the value.

The resulting first eight columns, that were generated, were the following: ID, ysoUri, term, lemma, isPartOfDocument, isPartOfSentenceNr, location, and hasWordClass. In addition to these columns, each possible type of syntactical relation (all 42 of them) had a column of their own. The Machinese Syntax analysis gives only one syntactical relation for every word. Therefore on each row there was always only one syntactical relation column that had a value. This value contained the localname of the resource of the term with which this word had a dependency with. The resulting data set contained 1432905 rows and 50 columns.

4.9 Relation Matrices

The hyponymy and associative relations from YSO were used to add ontology information to the terms of the documents. The matrix $\mathbf{A}_r$, $r \in \{$"hyponyms", "hypernyms", "hyponyms of hyponyms", ..., "associative relations"$\}$ of size $T \times T$ was defined, that could represent all desired relations. The matrix contains a binary representation of the r relation between the terms.

The attractiveness of this approach is that all possible extensions of the hyponymy relations can be created by simple multiplications and transposes of the matrix $\mathbf{A}_1$. Note that we did not use any relations $\mathbf{B}_s$ between the documents in these experiments.

5. EXPERIMENTAL RESULTS

The model with and without ontology information was trained with a set of 500, 1000 and 1500 documents. A dimensionality reduction using PCA was performed on all sets before training. The dimensionality was reduced to 20 and 50. Different kinds of ontology extensions were tested by adding hypernyms from 1, 2 and up to 3 levels. Overall 24 models were trained. Figure 3 shows the accuracy rates of those models when tested with a test set of documents that weren't in the training set.

The models with the best accuracy rate for a dimension and a data set size are marked with bold. The last two columns in the table titled MI 1 and MI 2 are the numbers of maximum improvement of the accuracy rate. MI 1 is the maximum improvement of the accuracy rate when ontology information is added and MI2 is the maximum improvement when using different kinds of ontology expansions.

At its best the accuracy rate of 74.18 % was reached with the model that was trained using a set of 1500 documents, was PCA reduced to 50 dimensions and used an ontology expansion of 2. In five out of six cases the maximum accuracy rate was reached by using an ontology expansion of 2, but the accuracy rate didn't variate that much between different kinds of ontology expansions. Especially as the training set grew, the maximum improvement on the accuracy rate by using different kinds of ontology expansions (see MI 2 column) became smaller.

At its best the accuracy rate improved by 5.96 % when adding ontology information to the model. The overall accuracy also always improved as the size of the training set grew.

6. RELATED WORK

Enhancing traditional machine learning techniques in the context of document analysis has been researched for around 15 years and a lot of the research on enhancing ML is done on hierarchical information, because the use of it amongst others enables for powerful generalisations [17]. For example two different documents, where one mentions only "pork" and the other only "beef", can easily be linked together, because from an ontology one can see, that the terms hypernym is "meat" [21].

Taskar et al. [32] introduced Relational Markov Networks and applied it to collaborative classification of related documents. Popescul et al. [27] used a relational model using both citation and coauthorship data. In collective classification, information about the predicted classes propagates over a network defined by the relations, see [29] for an overview. Our method differs from these in two ways: Firstly, we do not use the class information of related documents, only their observed term counts. Secondly, we also use the relations between terms.

Wittbrock et al. [34] from Cycorp, Inc. use geographical subsumption information from the company's own ontology to enhance location information for terrorist attack predictions. The probabilistic model they use benefits from the additional information. If an attack happened in Madrid, the probabilistic model can also comprehend that it happened in Spain and update the probabilities appropriately.

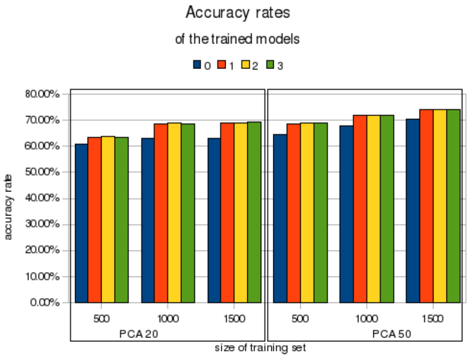
A hierarchical set of concepts that is repeatedly used to enhance the performance of traditional ML models is WordNet, and English lexical database. It contains words presented as pairs of a word's

[7] As of this writing (May 2010) the Semantic Computing Research Team has finished adding English labels to all the terms in YSO.

[8] `http://www.w3.org/TR/rdf-schema/#ch_literal`

$\mathbf{A}_r$	*relation*	*note*
$\mathbf{A}_0$	identity	Apple is an apple. $\mathbf{A}_0 = \mathbf{I}$
$\mathbf{A}_1$	hyponyms	Apple is the subclass of fruit, thus the term $a_{\text{apple,fruit}} = 1$
$\mathbf{A}_2$	hypernyms	transpose of $\mathbf{A}_1$
$\mathbf{A}_3$	associative relations	rain is associated with an umbrella thus the term $a_{\text{rain,umbrella}} = 1$ and $a_{\text{umbrella,rain}} = 1$
$\mathbf{A}_4$	hyponyms of hyponyms	$\mathbf{A}_1\mathbf{A}_1$
$\mathbf{A}_5$	hypernyms of hypernyms	$\mathbf{A}_2\mathbf{A}_2$
...	...	...

Table 1: The matrices for ontology information. If for example an ontology expansion two levels up and one level down want to be made plus associative terms want to be accounted for, then the matrices 1,2,3 and 5 come in question.



PCA	set size	accuracy rate with ontology extensions			
		none	1	2	3
20	500	60.87 %	63.39 %	**63.56 %**	63.46 %
	1000	62.99 %	68.68 %	**68.73 %**	68.68 %
	1500	63.11 %	69.06 %	69.05 %	**69.07 %**
50	500	64.61 %	68.60 %	**68.81 %**	68.79 %
	1000	67.88 %	71.91 %	**71.96 %**	71.83 %
	1500	70.26 %	74.08 %	**74.18 %**	74.12 %

Figure 3: Accuracy rates for models trained with a set of 500, 1000 and 1500 documents, PCA reduced dimensions of 20 and 50 and with ontology expansions of 0,1,2 and 3.

lexical form, the string representation, and its meaning. The concepts are linked with pointers that mark a semantic relation between them. The semantic relations can be synonymy, antonymy (the opposite of synonymy), hyponymy, meronymy, troponymy, which is the equivalent of hyponymy for verbs, and entailment, which marks an associative relation between verbs. The target of the semantic relations of a concept are packed in a set called a synset of the concept. [26]

Scott and Matwin [28], Rodríguez et al., Ureña-López et al. [33], and Hotho et al. [21] augment automatic document classification by using the synsets of WordNet in different ways to calculate the weights for various ML models.

[33] (and its earlier version [14]) use the Vector Space Model (VSM) to represent the text of documents and the categories, to which the documents need to be classified to. In their research only the categories are expanded with the synonyms of the synsets from WordNet. The automatic categorisation of documents improves from circa 50% to 65% on average.

[28] use WordNet synsets for expanding the text representation of the documents. They compare their approach to that of [14] and find that Rodríguez et al. approach isn't sufficient enough, as synonyms are picked manually. [28] add the synonyms and hypernyms of all verbs and nouns to the set of terms. As in this research they, too, tried out different levels of generalisation with hypernyms and found best results with an generalisation level of 2.

[21] expand the text representation with synsets from WordNet, as well. They test three different ways of augmenting the term weights by firstly adding up all terms weights of a word's synset's concepts to the word's term weight, secondly using only the first concept from the word's ordered synset, and by thirdly picking the concepts from the word's synset that at their best represent the document's context. The third approach together with an expansion of a terms hypernyms and hyponyms up to five levels seems in their research to create best results.

Zhang and Mao [35] also used relations between documents for classification. They first found communities (or clusters) in the document network and produced new features that describe how strongly each document belong to each community. These features could be used alongside the term counts.

Several authors [5, 8] have emphasised the importance of semantic analysis of terms when the number of training samples is small. Our experiments did not show the difference of importance changing with data set size, but on the other hand, we used dimensionality reduction that can be interpreted as latent semantic indexing, also in the comparison method. As future work, we could compare to the situation where dimensionality reduction is not used.

7. ACKNOWLEDGMENTS

This work was supported by the FinnOnto 2.0 project and by the Academy of Finland project 'Bayesian Machine Learning Algorithms for Discovering Relations with Latent Variable Models'.

8. REFERENCES

[1] J. Aitchison, A. Gilchrist, and D. Bawden. *Thesaurus Construction and Use: A Practical Manual*. Europa Publications, 4th edition, 2000.

[2] O. Alm. Tekstidokumenttien automaattinen ontologiaperustainen annotointi. Master's thesis, University of Helsinki, Department of Computer Science, September 2007.

[3] E. Alpaydin. *Introduction to Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2004.

[4] F. N. Archive. Asiankäsittelyjärjestelmiin sisältyvien pysyvästi säilytettävien asiakirjallisten tietojen säilyttäminen yksinomaan sähköisessä muodossa (Specifications for the

Permanent Storage of Information on Digital Documents to be Contained in Case Treatment Systems), 2005. Also available at `http://www.narc.fi/Arkistolaitos/pdf-ohjeet/akj_maarays.pdf`.

[5] R. Basili, M. Cammisa, and A. Moschitti. A semantic kernel to classify texts with very few training examples. *Informatica, an international journal of Computing and Informatics*, 2006.

[6] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. Owl web ontology language reference, 2004.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007.

[8] S. Bloehdorn, R. Basili, M. Cammisa, and A. Moschitti. Semantic kernels for text classification based on topological measures of feature similarity. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 06)*, Hong Kong, 2006.

[9] W. N. Borst. *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*. PhD thesis, University of Twente, Netherlands, 1997.

[10] D. Brickley and R. Guha. Rdf vocabulary description language 1.0: Rdf schema, 2004.

[11] F. Ciravegna, A. Dingli, D. Petrelli, and Y. Wilks. User-system cooperation in document annotation based on information extraction. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02)*, 2002.

[12] Connexor Oy. Machinese Linguistic Analysers, 2006.

[13] N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. *Journal of Intelligent Information Systems*, 18(2–3):127–152, 2002.

[14] M. de Buenaga Rodríguez, J. M. G. Hidalgo, and B. Díaz-Agudo. Using wordnet to complement training information in text categorization. In *Recent Advances in Natural Language Processing II*, volume 189, pages 353–364, 1997.

[15] S. Deerwester et al. Improving information retrieval with latent semantic indexing. In *Proceedings of the 51st Annual Meeting of the American Society for Information Science 25*, pages 36–40, 1988.

[16] Finnish National Archive. SäHKE-hanke, Abstrakti mallintaminen (Abstract Modelling of the SäHKE Project), 2005. Also available at `http://www.narc.fi/sahke/Aineisto/SAHKE-abstrakti-V2-koko.pdf`.

[17] E. Gabrilovich and S. Markovitch. Feature generation for text categorization using world knowledge. In *IJCAI'05*, pages 1048–1053, 2005.

[18] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5:199–220, 1993.

[19] N. Guerino and C. A. Welty. An overview of ontoclean. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, pages 151–172, 2004.

[20] S. Handschuh, S. Staab, and A. Maedche. Cream — creating relational metadata with a component-based, ontology-driven annotation framework. In *Proceedings of K-Cap 2001,Victoria, BC, Canada*, 2001.

[21] A. Hotho, S. Staab, and G. Stumme. Ontologies improve text document clustering. In *3rd IEEE International Conference on Data Mining*, pages 541–544, 2003.

[22] E. Hyvönen, K. Viljanen, J. Tuominen, and K. Seppälä. Building a national semantic web ontology and ontology service infrastructure—the FinnONTO approach. In *Proceedings of the 5th European Semantic Web Conference (ESWC 2008)*. Springer-Verlag, June 1-5 2008.

[23] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[24] C. D. Manning and H. Schütze. *Foundation of Statistical Natural Language Processing*. The MIT Press, 2000.

[25] F. Manola and E. Miller. Rdf primer, 2004.

[26] G. A. Miller. Wordnet: a lexical database for english. In *Communications of the ACM*, volume 38, pages 39–41, 1995.

[27] A. Popescul, L. H. Ungar, S. Lawrence, and D. M. Pennock. Statistical relational learning for document mining. In *Proceedings of IEEE International Conference on Data Mining (ICDM-2003)*, pages 275–282, 2003.

[28] S. Scott and S. Matwin. Text classification using wordnet hypernyms. In *Workshop—Usage of WordNet in Natural Language Processing Systems*, Montreal, Canada, 1998.

[29] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3), 2008.

[30] G. Siolas and F. d'Alch Buc. Support vector machines based on a semantic kernel for text categorization. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)*, volume 5, 2000.

[31] R. Studer, V. R. Benjamins, and D. Fendel. Knowledge engineering: Principles and methods. *IEEE Transactions on Data and Knowledge Engineering*, 25(1–2):161–197, 1998.

[32] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proc. Conference on Uncertainty in Artificial Intelligence (UAI02)*, pages 485–492, Edmonton, 2002.

[33] L. A. Ureña-López, M. Buenaga, and J. M. Gómez. Integrating linguistic resources in tc through wsd. *Computers and the Humanities*, 35(2):215–230, 2001.

[34] M. Witbrock, E. Coppock, R. Kahlert, and B. Rode. Cyc-enhanced machine classification. Technical report, Cycorp, Inc, 2009.

[35] D. Zhang and R. Mao. Extracting community structure features for hypertext classification. In *Proceedings of the 3rd IEEE International Conference on Digital Information Management (ICDIM)*, pages 436–441, London, UK, 2008.