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

D1.2.1 Dealing with unlabelled data

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

Abstract

In many machine learning problem domains large amounts of data are available but the cost of correctly labelling it prohibits its use for model training. For us especially relevant are large quantities of raw information available on the internet that present an interesting challenge of how to successfully exploit information hidden within it without first having to invest much human work into manually tagging it. There exist several methods for using a small initial set of labelled data together with a large supplementary unlabelled data pool in order to learn a better hypothesis than just by using the labelled information. In document classification, it was reported that the overall performance of such system has improved on many data sets, when using unlabelled data or asking the user for the labels of selected examples – active learning. We present several approaches to using unlabelled data in document classification.

This deliverable presents an overview of the state of the art in this field and provides working implementations of methods found to be useful on large textual domains. On average, less than half of usually required labelled samples are needed for the same classification accuracy.

Keyword list: unlabelled data, active learning, semi-supervised learning, machine learning

WP1
Prototype/Report PU
Contractual date of delivery: 31.12.2004
Actual date of delivery: 4.1.2004

SEKT Consortium

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

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

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

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

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

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

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

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

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

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

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

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

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

Executive Summary

In many machine learning problem domains large amounts of data are available but the cost of correctly labelling it prohibits its use for model training. For us especially relevant are large quantities of raw information available on the internet that present an interesting challenge of how to successfully exploit information hidden within it without first having to invest much human work into manually tagging it. There exist several methods for using a small initial set of labelled data together with a large supplementary unlabelled data pool in order to learn a better hypothesis than just by using the labelled information. In document classification, it was reported that the overall performance of such system has improved on many data sets, when using unlabelled data or asking the user for the labels of selected examples – active learning. We present several approaches to using unlabelled data in document classification.

This deliverable presents an overview of the state of the art in this field and provides working implementations of methods found to be useful on large textual domains. On average, less than half of usually required labelled samples are needed for the same classification accuracy.

Contents

1. Introduction

In the recent years, enormous amounts of information has become available – most notably unstructured and semi-structured textual data available from the Internet. In order for this information to be of greater use, more of its structure needs to be discovered – to enable automated processing and reasoning. One of the tools used for this is machine learning.

Supervised machine learning is a process of learning a function based on the given examples. The examples are provided as ordered pairs of objects (A, B) and the learning algorithm induces the function f: A → B based on some inductive bias (prior knowledge / assumptions) which is needed for any generalization to be possible. The resulting function can be further used to map previously unseen input objects onto unknown target values.

Since the data available can have a large complexity, this inherently means that we are dealing with complex functions – and learning complex functions requires many training examples. Examples with known target values (a.k.a. *labelled examples*) are however usually not directly available and need to be manually created/labelled, which can be a time-consuming and/or expensive process. In order to minimize this cost, a lot of research has been conducted in the area of using unlabelled examples to aid in the process.

Two different approaches and their mixtures will be presented here. One designed to minimize the required additional human effort for labelling examples (Active learning) and the other to work with a fixed set of labelled and unlabelled examples (Semi-supervised learning). Related work in the two approaches is described in Section 2 and Section 3. Our approach is described in Section 4 followed by the architecture of our system in Section 5. The report concludes with some ideas for future work in Section 6 and Appendix with the users guide for our software components.

2. Description of Active Learning and Related Work

Active learning has a tight link to the problem of 'experiment design' addressed in statistical literature. It is a generic term describing a special, interactive kind of a learning process. In contrast to the usual (passive) learning where the student is presented with a static set of examples that are then used to construct a model, active learning paradigm means the student can 'ask' the 'oracle' (e.g. domain expert, user,…) for a label of an example (see Figure 1).

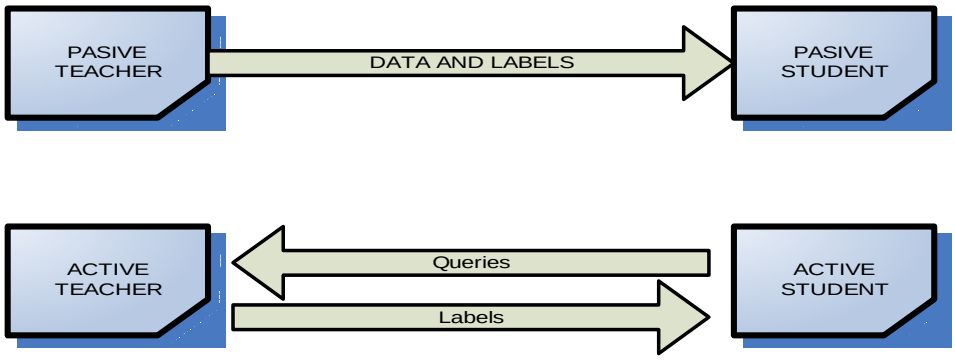

Figure 1. Illustration of passive versus active learning paradigm.

2.1. Query filtering approach to active learning

The intuition behind the paradigm of active learning is that having labels for a few highly informative examples provides much more information than having labels for many randomly chosen examples. (However, one must be careful since this violates the assumption of randomly sampled input made by many of the algorithms.) In general, one would expect from an active learning algorithm (the student in our case) to construct queries from scratch, which is difficult. For instance, in document classification, construction of a query means construction of a meaningful document to be labelled by an expert, and constructing it from a bag of words model commonly used for document classification is close to impossible. Since in many practical cases construction of queries used to obtain labels for the selected examples is hard, a 'query filtering' [1] paradigm becomes useful: the learning algorithm is

provided a large amount of non-labelled examples that are viewed as potential queries. Instead of a fixed set of examples, the input can also be a stream of possible queries and the algorithm is required to select the appropriate ones. Since the problem of finding the optimum subset of the most interesting questions is hard, a greedy approximation is used in practice, so that a sample selection is interleaved with asking questions. Knowing some of the answers before having selected all of the questions, in practice partially compensates for not selecting the optimal combination of questions. The basic active learning algorithm is the following:

```
Start with a small labelled set and a large set or a stream of unlabelled examples
repeat until some condition is met:
        from the unlabelled set select the currently most interesting example (or a batch of them)
        query the expert for the label
        add the now-labelled example to the labelled set
```

Algorithm 1. Active learning algorithm.

Another view to query filtering can be ‘starting form all the possible queries and discarding exponentially more and more possible queries for selecting only those whose answers information gain does not limit to zero’ if the input is a query stream.
The core of research of active learning algorithms is obviously the selection of the most interesting example. At the top level, there are two different approaches: indirect and direct model optimization.

2.2. Indirect methods for active learning

Indirect methods for active learning are derived from concept learning theory. A concept is a hypothesis with a Boolean target function – a sample is either a part of the concept or not. In order to learn a concept as fast as possible, we wish to minimize the version space (the set of all possible hypotheses that are still consistent with all of the examples seen so far) as fast as possible [2]. There exist theoretical proofs of exponential reduction in the number of required examples under certain assumptions [3], but the most limiting condition is that there is no noise present in the data. Using the real-world datasets it is often impossible to find even one hypothesis that fits the data. Therefore, it is not obvious why such an approach would successfully minimize the resulting error of the model when used on unseen examples.

These approaches can be divided to single- and multi-model based approaches. The multi-model approach is based on the idea that since the analytic form of version space bisection is generally not possible, we approximate it by infinitely many models randomly sampled from the version space. We should select the example with the highest prediction entropy across all of the sampled models. Such an example will on average remove the largest possible portion of the version space after being labelled. The idea is called "query by committee" or QBC [2]. We provide the basic algorithm in Algorithm 2. In practice, it of course requires another simplification step – the number of sampled models is finite and often quite small, with little degradation of accuracy. If performed on a stream of examples it should have even number of models and select only those queries where the predictions are split exactly in half. When using it on a fixed set of unlabelled samples the affinity for a sample should be weighted by the estimated density distribution of the data so that outliers receive less attention.

```
while not end of stream
        generate 2n models from the version space consistent with currently labelled examples
        predict the value of the current example
        if n examples predict positive and n examples predict negative
                query user
```

Algorithm 2. Query by committee algorithm.

With single-model methods [4, 1] another simplification is made: an assumption that a high certainty prediction by a single model also means that a large portion of models from the current version space would give the same prediction - meaning that after inclusion of that example into the

labelled learning set, only a small amount of hypotheses would be removed; therefore making that example inappropriate if the goal is to minimize version space as quickly as possible. Examples with *low* prediction certainty are then used to query the oracle. Algorithm 3 provides an outline of that algorithm.

Example selection step:

using currently labelled examples train a model that can output a prediction certainty (e.g. naïve bayes with bagging)
for each unlabelled example still available
predict the target value and remember the classifier certainty
pick N examples with the lowest certainty and submit them for classification

Algorithm 3. Uncertainty sampling algorithm.

The obvious problem with this method is that the aforementioned assumption is generally false. On the positive side the algorithm is relatively fast compared to other AL algorithms as there is no need for version space sampling.

An extension of previous ideas is an SVM-specific algorithm [5] called SVM margin minimization. For each example, two different models are built – one with the example temporarily put in the positive class and the other with example in the negative class. The example with the most similar margin sizes is selected for labelling, thus approximating version space bisection for SVM. This method however has a large time complexity: for each example considered, a couple of SVM models must be created as opposed to just evaluated as in uncertainty sampling. A simplification that selects the example only based on its distance from the current hyperplane can be made. This way only one model per sample (or a batch of them) needs to be created. On typical text domains, its accuracy is statistically equal to the original version for a fraction of the computational cost.

2.3. Direct optimization

The direct approach does not make the assumption that the data is noise-free. It does not try to minimize the version space but instead tries to directly minimize the expected future prediction error E of the final model over the entire sample space and so directly optimizes the criteria function with which the model will be evaluated – using some loss function L:

$$E_{P_D} = \int_x L(P(y \mid x), P_D(y \mid x))P(x)$$

Where L is the loss function (the disagreement between the models prediction distribution and the real class distribution), P(x) probability of a sample x, P(y|x) the true probability distribution of the label y for sample x and P_D the predicted probability. At each step such an example is selected that would – if added labelled – minimize the expected error. Of the many possible loss functions the log-loss is most commonly used:

$$L(P, P^*) = \sum_{y \in Y} P(y \mid x) \log(P^*(y \mid x))$$

where Y is the set of all possible labels and P and P^* real and predicted class probability density functions. Since the correct labelling is unknown, an approximation using the current model is again used. One possible approximation is thus to select such an example that results in a model with the minimum average prediction variance averaged over all of the possible target values [6]. Another option is to minimize the expected loss over a validation subset generated from the existing labelled data [7, 23].

The problem with the first algorithm is that it is using its own predictions to estimate its error and in turn reinforcing its own belief more and more. The latter algorithm performs very well – if there are many labelled examples - but that is generally not the case with active learning.

2.4. Summary

The performance of the aforementioned techniques (measured by the expert labelling cost) can vary from problem to problem by orders of magnitude. Computational cost should also be taken into account when choosing an approach - while decreasing the cost of human labour the CPU requirements can increase beyond any reasonable limit: in the usual learning scenario, one only needs to train one model, which can already be an expensive procedure. For a simple uncertainty-based active learning, one has to train the same number of models as there are labelled examples at the end use every one of them to test each unlabelled sample. It is possible to decrease the amount of CPU work by a constant factor at the expense of some human labour by selecting several examples at the same time without updating the rest of the system. For the method based on SVM margin sizes, the number of trained and discarded models is for each iteration of active learning loop linearly dependant on the size of the unlabelled set; making efficient implementation of incremental learning algorithms an absolute must. It should also be noted that there is no generic way of deciding which algorithm to use and how to set its parameters without first knowing the cost of human work compared to the computational cost.

3. Description of Semi-supervised Learning and Related Work

While also dealing with unlabelled data, semi-supervised learning [8] is not an interactive procedure. The system is provided with a set of labelled examples and a set of unlabelled examples, which can be used as an addition to gain an insight on the data. One possible use of unlabelled examples is to provide the estimation and correction of the sampling bias if the labelled examples have been sampled non-randomly [9].

3.1. Semi-supervised transduction

The first possibility in semi-supervised learning is transduction: one only has to label the already known unlabelled data and no model for later use needs to be constructed. One common approach is to first construct a graph using all of the examples as vertices and connect those vertices that are similar – close to each other according to a chosen distance measure – and assign that distance as the weight of the edge. That way an implicit bias based on the neighbourhood relationship is introduced into the model which provides the required background knowledge for the algorithm to be able to make use of unlabelled data. Labels from the labelled examples are then propagated to the unlabelled ones.

The simplest algorithm for assignment of binary labels is based on graph mincut [10], see Algorithm 4 for its outline. Since there is a possibility for the cuts in Algorithm 4 to be degenerated (e.g. if the graph is a path with all of the edge weights equal there are n-1 possible cuts but the mincut algorithm will return one of the cuts with one vertex on one side and the rest on the other – which is clearly not a desirable solution) an randomized version of the algorithm exists.

construct a graph using all of the examples
add two more vertices (one for each label) (+), (-)
connect labelled vertices with the corresponding (+) or (-) vertex with edges of infinite weight
connect the rest of the vertices with edges weighted by the similarity function
find the minimum cut between (+) and (-) thus minimizing the number of similar vertices that will be given different labels
assign labels to the unlabelled vertices depending on which side of the cut they are

Algorithm 4: Min-cut transduction algorithm.

3.1.1. Learning using Gaussian Fields and Harmonic Functions

The idea behind this method of Gaussian Fields and Harmonic Functions [24] is as follows. The data points that lie nearby should have similar labels. If the evaluation of a labelling is

$$E(f) = \frac{1}{2}\sum_{i,j} w_{ij}\left(f(x_i) - f(x_j)\right)^2$$

where w_{ij} is the similarity between the two data points x_i and x_j, f is the value of a sample, it is possible to show that the optimal value f at each data point is

$$f(x_j) = \frac{1}{d_j} \sum_{i \sim j} w_{ij} f(x_i)$$

for j=[labeled+1, …, labeled+unlabelled] and d_j the sum of all w_{ij} for that j; which is the weighted average of its neighbours labels. Instead of an iterative approach of label propagation, the labels can be computed in a single step as

$$f_u = (I - P_{uu})^{-1} P_{ul} f_l$$

where

$$P = D^{-1}W, f = \begin{bmatrix} f_l \\ f_u \end{bmatrix}$$

with W being similarity matrix, D the diagonal matrix with elements as sums of W's columns and P_{uu} and P_{ul} appropriate parts of the matrix P (u-unlabelled, l-labeled), analogous to vector f.

3.1.2. Learning with local and global consistency

As proposed in [25], instead of the previous evaluation function we can accept that even the values of the already labeled examples can change as a tradeoff for better local consistency. That gives the equation

$$Q(F) = \frac{1}{2} \left(\sum_{i,j=1}^{n} W_{i,j} \left\| \frac{1}{\sqrt{D_{ii}}} F_i - \frac{1}{\sqrt{D_{jj}}} F_j \right\|^2 + \mu \sum_{i=1}^{n} \left\| F_i - Y_i \right\|^2 \right)$$

where W and D are the same as before and μ the regularization parameter, Y a matrix of `(N_labeled + N_unlabelled) x N_labels` dimensions with Y_{ij}=1 if i-th labeled example belongs to the class j, otherwise 0 and F the output matrtix with the same properties and classification function

$$F^* = \arg\min Q(F)$$

The first term of the equation requires similar labels of nearby samples (a.k.a. smoothness constraint) and the second term is a fitting constraint (the predicted values should not differ much from the real ones). An analytical solution can again be computed, giving

$$F^* = (I - \alpha S)^{-1} Y$$

where $S = D^{-1/2} W D^{-1/2}$.

All of the described methods have the largest benefit where there is only a *very* small amount of labeled examples; with larger datasets problems such as feature relevance (that should be used in calculation of similarity matrix) become obvious.

3.2. Semi-supervised induction

Semi-supervised inductive methods are mostly based on expectation maximization (EM) – an iterative algorithm for improving the hypothesis. The general idea is to create the initial model, label the unlabelled data and then iteratively generate a new model using all of the labels, re-label the originally unlabelled data using the resulting model; stopping when some convergence criteria is met (see Algorithm 5). The problem with this approach is that a lot of incorrectly labelled examples in the initial steps of the algorithm mask the labelled examples, forcing the model to converge to a random point [8]. A weighting of the samples must therefore be used. There is no definitive way of deciding the importance of the unlabelled samples; usually it is gradually increased from 0 to 1. A separate

validation set can be used to provide some guarantees about the real accuracy, but that increases the labelling effort since the validation set needs to be labelled first. Given a validation set, it is also possible to vary the weighting scheme and at the end only use the best results. Alternative solutions have also been proposed [16] but are much too complex for practical use on large domains.

learn the initial model on the labelled samples
use it to probabilistically assign labels to unlabelled samples
rebuild model using new labels of ALL samples
repeat from step 1 until some convergence criteria met

Algorithm 5. EM model induction algorithm.

3.2.1. Co-training

In the case that multiple independent views (i.e. two or more mutually *independent* sets of attributes describing the same examples) of data are available maximization of prediction consistency across models trained on different views can be attempted. The 'co-training' [17] algorithm iteratively learns multiple models (one on each view) and allows each of them to label some unlabelled examples. The examples with the most confident prediction are then added to the labelled set and the process is repeated.

Even when the views on the data are not completely independent, the algorithm still has a better accuracy compared to the model simply trained on the union of all the features. It was shown that the independence of views is unnecessarily strong condition and that even a random feature split (all of the features randomly split in 2 or more segment partition, each used as input for one model) can be used as input for a co-training algorithm [26]. However, due to the already mentioned problems with EM, running such an algorithm on textual input requires aggressive feature selection and is much more stable with large starting numbers of labelled input samples.

The Co-EM [18] algorithm combines EM and co-training. It uses the hypotheses learned from one view to probabilistically label the examples which are then used to learn a hypothesis on another view.

3.3. Constrained clustering

Constrained clustering is clustering with background knowledge. Constraints can be instance based [19] (e.g. two examples must / must not be in the same cluster), hard (mandatory) or soft (unobserved constraints add penalty), global constraints [20] (e.g. each cluster must have at least N elements) or even in a form of declarative knowledge (a subset of FOL in [21]). Conversion of ordinary iterative clustering algorithms into constrained clustering is quite straightforward by obeying the constraints at every class reassignment step. The constraints even reduce the size of the search space, making the algorithms faster than their counterparts without them.

4. Description of the chosen approach

Based on the published results we have chosen to implement the Simple Margin method described in [5], active learning with sampling estimation of error reduction [7] and an uncertainty sampling using an ensemble of naïve Bayesian classifiers. All of the methods are provided with two input pools of labelled and unlabelled document vectors in TFIDF format. TFIDF is a simple representation of a document from the 'bag-of-words' familiy. Each document is represented as a (usually sparse) vector in a space of all words we wish to use; each dimension signifying the presence and amount of a certain word in the document. Because some words are more meaningful than others, term frequency (TF) can be weighted by the inverse of the number of documents in which the word (term) occurs (inverse document frequency – IDF) – giving TF*IDF or simply TFIDF. For a detailed explanation see [29]. In each iteration, the algorithms are allowed to return up to a predefined constant number of queries (indices of unlabelled documents). With each query a decimal number signifying the estimated importance of the query is also submitted in order to enable graphical user interfaces to order and collate multiple related queries to further ease the labelling work. The Simple Margin algorithm creates a linear SVM based on currently labelled examples in each iteration of the loop. It then calculates the distance of all of the unlabelled samples from the hyperplane, orders them by ascending order and selects first N to be sent for user labelling.

The expected error estimation method is implemented using the log-loss error measure. Because of the 'naïve' assumption of the naïve Bayesian models, they have a very sharp posterior class distribution, making them unsuitable for direct use for uncertainty estimation. This problem can be somewhat minimized by using bagging [27] approach to model building. Instead of one, many almost identical copies of the model are built using the same data source, but instead of giving each the entire input, the input is sub-sampled (with replacement) – creating a 'bag' of models with slightly different resulting predictions. The predictions are then averaged together making the final prediction distribution smoother. Unfortunately, for the small numbers of models in the bag (<30) the predictions are still too sharp for use in error estimation; and for large numbers the procedure becomes computationally inefficient. Using bagged models in uncertainty sampling gives only slightly better results than random sample selection. Because of this, an algorithm for converting SVM predictions into probabilistic output by fitting a sigmoid function to the predictions, described in [28] was used to replace the bagged naïve Bayesian model in the 'Sampling estimation of error reduction' with a linear SVM.

In each iteration of 'Sampling estimation of error reduction', a model is first trained on the known data. Two smaller sets are sampled from the unlabelled data pool, one for query candidates and one for evaluation set. It would be possible to use the entire unlabelled pool but the method is already much slower than both other – with the query pool size of 50 it is about 40 times slower than Simple Margin. Each eligible query is then evaluated for expected reduction of model error and the best few are selected for submission to the user. Because of the lack of a true test set, an approximation is used as described under the related work section. For each possible query two models are created, one with the query added to the positive and the other to the negative class. The entropy of all possible classifications of the resulting model over the evaluation set is calculated and weighted by the current probability of the assumed class of the tested query example.

The described methods were compared to each other on a standard text domain (Reuters news corpus volume 1 – RCV1) in a single category (versus everything else) classification task. The RCV1 is a collection of about 800.000 news articles from Reuters Ltd., categorized into a small taxonomy of about 100 categories such as "Economics", "Construction", "Advertising", etc. As expected, all of the methods consume much more CPU time than a standard supervised learning; but even among the AL algorithms there are large differences. Fortunately, one of the least expensive algorithms – the SVM margin minimization, achieved the best classification accuracy (on text) as can be seen from the figures 1 and 2. It has a consistently better accuracy than the model using the same number of randomly selected and labelled examples. In some cases it even exceeds the accuracy achieved by a model that has all of the examples labelled – meaning that it successfully 'throws away' data that only confuses it. This phenomenon needs to be further studied and a robust stopping criterion devised – since the performance of the model begins to decrease when it is forced to learn on non-informative data. The classification accuracy of the simple margin minimization approach is almost equal to the one achieved by training two SVM models for each sample considered and estimating the version space reduction as the ratio between their margins sizes - for at least one magnitude smaller computational cost - therefore the former approach was selected for the final implementation. We based the implementation on the standard (non-incremental) SVM training algorithm (described in

Deliverable D1.5.1) because almost every new added sample is a support vector – since it is the one closest to the learned hyperplane – and therefore introduces too large modifications of the hyperplane. Other existing active learning algorithms (such as active learning with estimation of error reduction) are much less stable on high-dimensional textual data, are too CPU expensive or have worse performance than the chosen algorithm. An example of such an algorithm is uncertainty sampling using bagged Naïve Bayes model.

The following figures show the classification performance of the models built using an active learning approach compared to a baseline model built with the same number of randomly selected data points. The 'Sampling estimation of error reduction' was not tested for larger datasets because of its complexity.

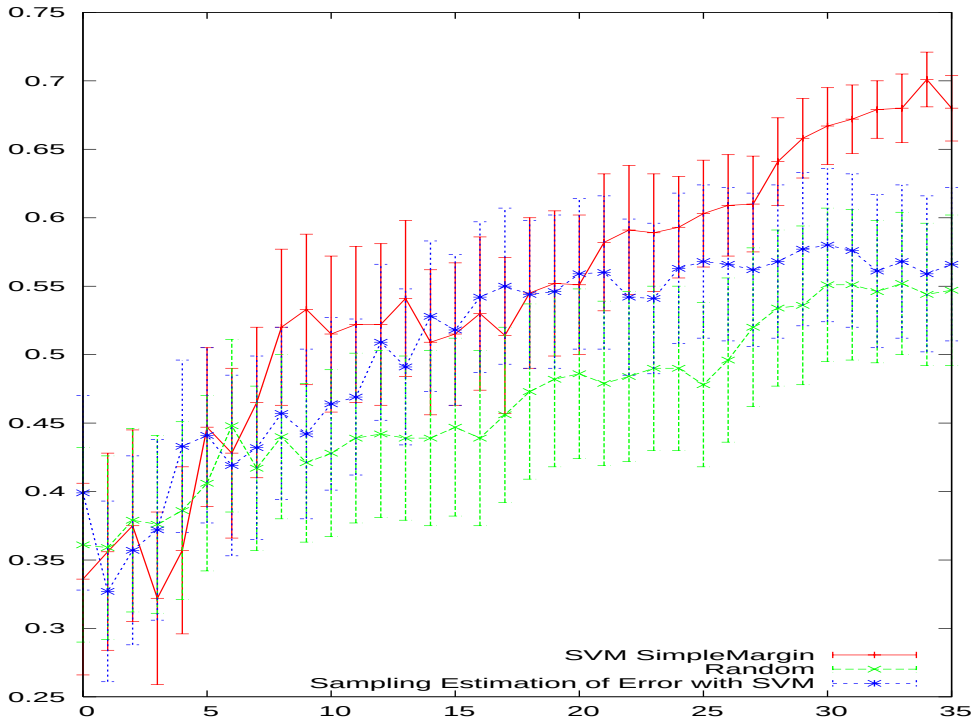

Figure 2. Performance of best active learning methods on small datasets (average F_1 measure as a function of number of additionally labelled samples, starting with 10 randomly pre-labelled)

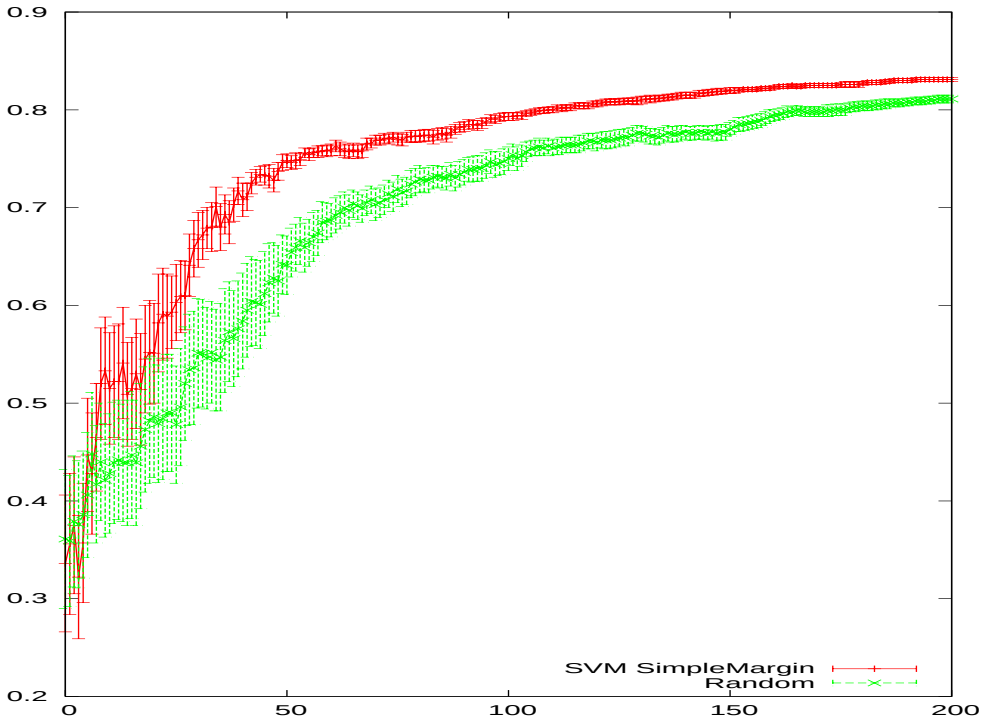

Figure 3. Performance of best active learning method on larger dataset (average F_1 measure as a function of number of additionally labelled samples, starting with 10 randomly pre-labelled)

For semi-supervised approach, three different methods were used for the final implementation: co-training, and the methods described in [24] ('*Semi-Supervised learning using Gaussian Fields and Harmonic Functions*') and [25] ('Learning with local and global consistency'). They work much better for low-dimensional problems than for textual inputs, so they will be adapted for future machine learning problems in SEKT as the need arises. (When using very small labelled datasets (5-35 labeled samples) in conjunction with a large unlabelled pool, the classification accuracy of semi-supervised methods compared to SVM and weighted nearest neighbour based transduction can be as much as 5.5 times better on a problem domain with only 256 dimensions compared to only 20% improvement on a text domain with ~10.000 dimensions.)

5. Architecture

The architecture of the active learning system is provided in Figure 5. The input to the system is a set of labelled documents and usually a much larger set of unlabelled documents. The output is a Support Vector Machine model that can be further used for document classification.

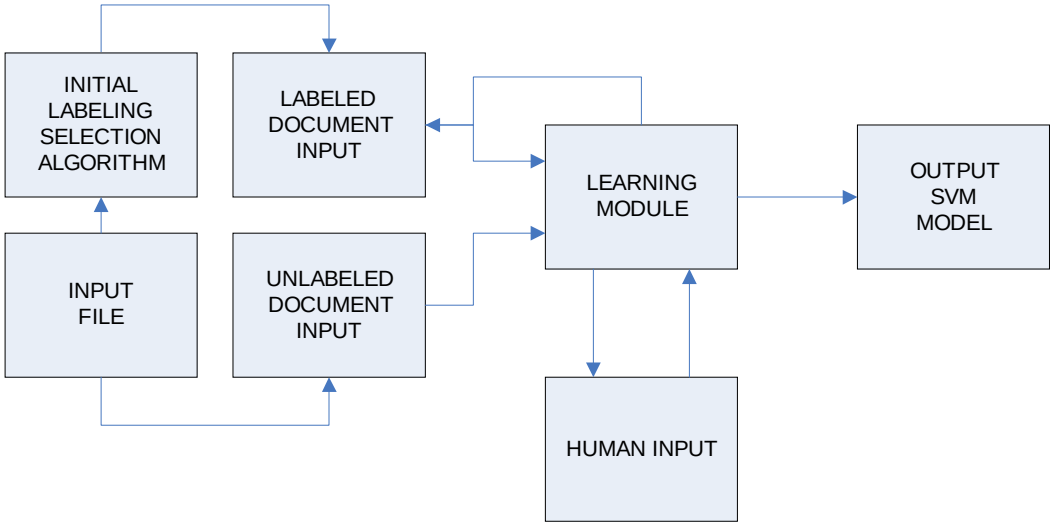

Figure 4. Active learning architecture as implemented in TextGarden

Because of the inherent interactive nature of active learning, all of the active learning binaries use the standard input/output for communication with the user. The communication protocol is line based and asynchronous. Every line written to standard output is a query and has the form

```
QUERY_ID "MEM" DOCUMENT_ID
```

or

```
QUERY_ID "IS-A" DOCUMENT_ID CURRENT_PROB CATEGORY_ID
```

and the answer for both query types read from standard input has a form of

```
QUERY_ID CATEGORY_ID*
```

or an

```
"END"
```

literal; where QUERY_ID is an ASCII representation of an unsigned integer used for associating answers with queries, DOCUMENT_ID is the TextGarden document base ID of a document, CATEGORY_ID likewise for categories, CURRENT_PROB the estimated probability of the positive answer (which can be used for question ordering by the user interface), "MEM" a literal string representing a full membership query and "IS-A" a simple membership query – the difference being that only zero or one category ID is expected for a simple query (meaning the document belongs to that category) whereas the full membership query requires an ASCII space separated list of all of the categories the document belongs to. The "END" string instructs the program to finish immediately and write the current model to the output file – even if not all of the input samples were queried for.

Examples:

```
Q: 001 MEM 17        // what categories does document 17 belong to?
A: 001 3 5 6         // it belongs to 3, 5 and 6
Q: 002 IS-A 18 5     // does document 18 belong to category 5?
A: 002 5             // yes
Q: 003 IS-A 17 7     // does document 17 belong to cat. 7?
A: 003               // no
A: END               // write output and quit
```

The architecture of the semi-supervised system is provided in Figure 6. The input to the system is a set of labelled documents and usually a much larger set of unlabelled documents. The output is a labelled document set consisting of potentially relabelled documents form the labelled set and some of the unlabelled labelled documents with added labels. Such labelled document set is then further used in learning in the same way as manually labelled documents.

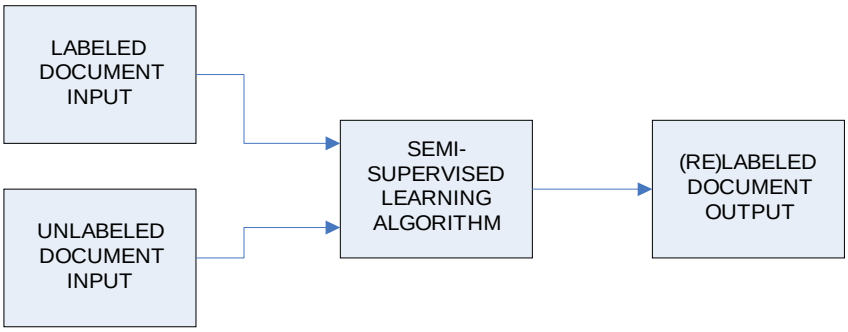

Figure 5. Semi-supervised transduction architecture as implemented in TextGarden

6. Conclusion and Future Development

The main result of the research of use of unlabelled data is the implementation of the selected active learning algorithms. The most important practical consequence of using it is the direct decrease of human work needed to label the learning examples to only a small fraction of the full dataset – often down to only 40%-50% of randomly selected dataset.
In SEKT, we will use these methods mainly for semi-automatic ontology learning and population. In ontology population, a large number of specimens need to be inserted into the ontology and the computer can help by only asking questions that will have an impact on the result. For semi-automatic ontology learning, the insight on the data acquired by the learning algorithm can guide the user to the final version of the ontology by a shorter path than the completely manual process. The described methods can be further used in SEKT tasks on document annotation inside the tool GATE, to help the user in providing labelled examples that can be further used in training a classifier.
Combining active learning and semi-supervised methods for practical use is still a very unresearched area. The improvements are also possible in uncertainty calculation and optimizing the time performance of the algorithms. Yet another possibility is the consideration of n-tuple combinations of samples for queries that have a larger information gain together than the sum of each. The algorithms will also be extended to work on other problem domains. This will be done on a case per case basis when data becomes available within the SEKT project.

8. Appendix - User Guide

Active learning on sparse training sets using binary SVM model

The utility *ALTrainBinSVM.exe* performs active learning loop on the specified input. The input is a set of unlabelled vectors in the form of a sparse trainset (".sts") file. The ".sts" file is a binary serialization of TextGarden data structures; other TextGarden utilities can be used to transform various input data formats (e.g. Bag-Of-Words) into a 'sparse trainset'. Until sufficient number of samples for both classes are acquired, labeling requests are created randomly or by using a smart selection based on the data structure. When enough examples are labeled for the SVM to converge to a meaningful result, the SimpleMargin algorithm is used to calculate the next batch of queries. The size of the batch sets the number of required answers before the main loop is run again. The best results are usually achieved by a batch size of 1 but at the same time this setting requires the most CPU time (which is inversly proportional to the batch size). When the prespecified number of queries is processed, the user returns an "END" literal or the unlabelled pool is depleted the program terminates with an exit code of 0 and writes the resulting SVM model, the union of the labeled and unlabelled pool or both to the specified locations.

usage: ALTrainBinSVM.exe
-i:Input-SparseTrainset-Data (default:'')
-b:Batch-Size (default:1)

-s:Initial-Selection-Mode (default:0) (0:random, 1:data analysis)
-q:Max-Queries (default:-1)
-do:Data-Output-File (default:'')
-mo:Model-Output-File (default:'')

Example:

```
ALTrainBinSVM.exe –i:input.sts –b:2 –s:1 –q:100 –mo:test.svm
[communication on standard input/output]
> 001 IS-A 43 0.501 1
< 001 1
> 002 IS-A 2231 0.447 -1
< 002
< END
…
```

Semi-Supervised transduction

SSTransduction.exe utility performs a transductive inference on a joint labeled and unlabelled dataset. The input is a set of unlabelled vectors in the form of a sparse trainset (".sts") file. The vector parameters in the input file are assumed to be 0.0 for 'unlabelled' and positive integers 1..N for classess 1 to N. The result is written to the output file with the same vector ordering.

usage: SSTransduction.exe
-i:Input-SparseTrainSet-Data (default:'')
-o:Output-SparseTrainSet-Data (default:'')

REFERENCES

[1] "A sequential algorithm for training text classifiers", David D. Lewis and William A. Gale. Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval.
[2] "Query by Committee", H. S. Seung and Manfred Opper and Haim Sompolinsky. Computational Learning Theory pg. 287-294, 1992.
[3] "Information, prediction, and query by committee", Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Advances in Neural Information Processing Systems 5, pages 483-490, 1993.
[4] "Improving Generalization with Active Learning", David A. Cohn and Les Atlas and Richard E. Ladner. Machine Learning 15-2, pg. 201-221, 1994.
[5] "Support Vector Machine Active Learning with Applications to Text Classification", Simon Tong and Daphne Koller. Proceedings of ICML-00, 17th International Conference on Machine Learning, pg. 999-1006.
[6] "Active Learning with Statistical Models", David A. Cohn and Zoubin Ghahramani and Michael I. Jordan. Advances in Neural Information Processing Systems vol 7, pg. 705-712, 1995.
[7] "Toward Optimal Active Learning through Sampling Estimation of Error Reduction", N. Roy, A. McCallum. Proceedings of 18th International Conference on Machine Learning, 2001, pp 441-448.
[8] "Learning with Labeled and Unlabeled Data", Matthias Seeger. Technical Report, Edinburgh University, 2000.
[9] "Automatic Bayes Carpentry Using Unlabeled Data in Semi-Supervised Classification", H. Zou, J. Zhu, T. Hastie. http://citeseer.ist.psu.edu/669591.html
[10] "Learning from Labeled and Unlabeled Data Using Graph Mincuts", Avrim Blum and Shuchi Chawla. Proc. 18th International Conf. on Machine Learning, pg. 19-26, 2001.
[11] "Semi-Supervised Learning Using Randomized Mincuts", A. Blum, J. Lafferty, M. R. Rwebangira, R. Reddy. In Proceedings of the 21st International Conference on Machine Learning, 2004.
[12] "Transductive Learning via Spectral Graph Partitioning", Thorsten Joachims. In Proceedings of the Twentieth International Conference on Machine Learning, 2003.
[13] "Learning from Labeled and Unlabeled Data Using Random Walks", D. Zhou, B. Scholkopf. DAGM'04: 26th Pattern Recognition Symposium (2004)

[14] "Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions", Xiaojin Zhu, Zoubin Ghahramani, John Lafferty. In Proceedings of The Twentieth International Conference on Machine Learning, 2003.
[16] "Stable Mixing of Complete and Incomplete Information", Adrian Corduneanu and Tommi Jaakkola. AI Memo 2001-030.
[17] "Combining Labeled and Unlabeled Data with Co-training", Avrim Blum and Tom Mitchell. COLT: Proceedings of the Workshop on Computational Learning Theory, 1998.
[18] "Analyzing the Effectiveness and Applicability of Co-training", Kamal Nigam and Rayid Ghani. In Proc. of Ninth International Conference on Information and Knowledge (CIKM-2000), pp 86-93.
[19] "Clustering with Instance-level Constraints", K. Wagstaff, C. Cardie. In Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, 2000, pp 1097.
[20] "Constrained K-Means Clustering", P. S. Bradley, K. P. Bennett, A. Demiriz. Microsoft Research technical report MSR-TR-2000-65.
[21] "Integrating Declarative Knowledge in Hierarchical Clustering Tasks", L. Talavera and J. Bejar. Lecture Notes in Computer Science vol. 1642, 1999; pp 211+.
[22] "Active + Semi-Supervised Learning = Robust Multi-View Learning", I. Muslea and S. Minton and C. A. Knoblock. In Proceedings of the Nineteenth International Conference on Machine Learning, 2002; pp 435-442.
[23] "Active Sampling for Class Probability Estimation and Ranking", M. Saar-Tsechansky and F. Provost. Machine Learning, vol. 54, issue 2, 2004, pp. 153-178.
[24] "Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions", Xiaojin Zhu, Zoubin Ghahramani, John Lafferty. In Proceedings of the 20th International Conference on Machine Learning (ICML-2003).
[25] "Learning with Local and Global Consistency", Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston and Bernhard Schölkopf. In 18th Annual Conf. on Neural Information Processing Systems, 2003.
[26] "Co-training with a Random Split of a Single Natural Feature Set", Jason Chan, Irena Koprinska, Josiah Poon.
[27] "Bagging predictors", Leo Breiman. Machine Learning vol. 24, 1996.
[28] "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods", John C. Platt. In: Advances in Large Margin Classifiers, MIT Press.
[29] "Developments in Automatic Text Retrieval" G.Salton. Science, Vol 253, pages 974-979, 1991