Proceedings of the ICML 2005 Workshop on

Learning With Multiple Views

Bonn, Germany, August 11th, 2005

Stefan Rüping and Tobias Scheffer (Eds.)

Contents

Foreword

Multi-view learning is a natural, yet non-standard new problem setting; it describes the problem of learning from data represented by multiple independent sets of features. A typical example is learning to classify web pages by either the words on the page or the words contained in anchor texts of links to the page. Multi-view learning methods have been studied under different names by de Sa (1994), Yarowsky (1995), Blum and Mitchell (1998), Dasgupta et al. (2001), and Abney (2002) among others. Their results suggest that there may be an underlying principle which gives rise to a family of new methods: A high consensus of two independent hypotheses results in a low generalization error.

In the last 2-3 years, several new supervised and unsupervised methods have been proposed which utilize this consensus maximization principle in one way or another. However, in many cases the contributors seem to be not to the full extent aware of the relationships between their methods and a possible common underlying principle.

This volume contains the contributions to the Workshop on Learning with Multiple Views, held at the 22nd International Conference on Machine Learning in Bonn, Germany. Contributions to this workshop from fields of machine learning, such diverse as clustering, semi-supervised learning, named entity recognition or ensemble learning show that there is a strong interest in learning problems with multiply represented instances and consensus maximizing learning methods in in a variety of communities. We hope that this workshop can help to make the intrinsic structure of this field becomes more clearly visible and to bring this interesting and rapidly developing area to the attention of additional researchers.

We would like to thank the members of the program committee, the organizers of the ICML, in particular Hendrick Blockeel, and everyone else who helped us in preparing this workshop for their kind support.

Stefan Rüping and Tobias Scheffer

Optimising Selective Sampling for Bootstrapping Named Entity Recognition

Markus Becker M.BECKER@ED.AC.UK
Ben Hachey BHACHEY@INF.ED.AC.UK
Beatrice Alex V1BALEX@INF.ED.AC.UK
Claire Grover GROVER@INF.ED.AC.UK
School of Informatics, University of Edinburgh, 2 Buccleuch Place, Edinburgh, EH8 9LW, UK

Abstract

Training a statistical named entity recognition system in a new domain requires costly manual annotation of large quantities of in-domain data. Active learning promises to reduce the annotation cost by selecting only highly informative data points. This paper is concerned with a real active learning experiment to bootstrap a named entity recognition system for a new domain of radio astronomical abstracts. We evaluate several committee-based metrics for quantifying the disagreement between classifiers built using multiple views, and demonstrate that the choice of metric can be optimised in simulation experiments with existing annotated data from different domains. A final evaluation shows that we gained substantial savings compared to a randomly sampled baseline.

1. Introduction

The training of statistical named entity recognition (NER) systems requires large quantities of manually annotated data. Manual annotation however is typically costly and time-consuming. Furthermore, successful application of NER is dependent on training data from the same domain. Thus, bootstrapping NER in a new domain typically requires acquisition of new annotated data. Active learning promises to reduce the total amount of labelled data by selectively sampling the most informative data points.

We introduce the newly created Astronomical Bootstrapping Corpus (ABC), which contains abstracts of radio astronomical papers, and report on our assessment of active learning methods for bootstrapping a statistical named entity recognition (NER) system for this new domain.

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

As part of our methodology, we experimented with a NER system in a known domain with existing corpus resources, namely the Genia corpus of biomedical abstracts (Kim et al., 2003). We tested relevant active learning parameters in simulation experiments with a view to arrive at an optimal setting for a real active learning experiment in the new astronomical domain. This was of particular importance since we were budgeted only 1000 sentences for active learning annotation.

We employ a committee-based method where trained classifiers are caused to be different by employing multiple views of the feature space. The degree of deviation of the classifiers with respect to their analysis can tell us if an example is potentially useful. We evaluate various metrics to quantify disagreement and demonstrate that the choice of metric can be optimised in simulation experiments with existing annotated data from distinct domains.

In the following section, we present the new corpus of astronomy abstracts developed for the bootstrapping task. In section 3, we introduce our active learning set-up for bootstrapping named entity recognition. Next, section 4 contains experimental results for a series of simulated active learning experiments used for parameter optimisation and section 5 contains the bootstrapping results. Finally, section 6 contains conclusions and future work.

2. The Corpus

2.1. Astronomical Named Entities

The main purpose of the corpus development work was to provide materials for assessing methods of porting a statistical NER system to a new domain. To do this we needed to create a small annotated corpus in a new domain which would serve as a basis for experiments with bootstrapping. Our chosen new domain was abstracts of radio astronomical papers and our corpus consists of abstracts taken from the NASA Astrophysics Data System archive, a digital library for physics, astrophysics, and instrumentation (http://adsabs.harvard.edu/preprint_service.html).

On the Column Density of <**Source-type**>AGN</**Source-type**> Outflows: The Case of <**Source-name**>NGC 5548</**Source-name**>

We reanalyze the <**Instrument-name**>Hubble Space Telescope</**Instrument-name**> high-resolution spectroscopic data of the intrinsic absorber in <**Source-name**>NGC 5548</**Source-name**> and find that the <**Spectral-feature**>C IV absorption</**Spectral-feature**> column density is at least 4 times larger than previously determined. This increase arises from accounting for the kinematical nature of the absorber and from our conclusion that the outflow does not cover the narrow emission line region in this object. The improved column density determination begins to bridge the gap between the high column densities measured in the X-ray and the low ones previously inferred from the <**Spectral-feature**>UV lines</**Spectral-feature**>. Combined with our findings for outflows in high-luminosity <**Source-type**>quasars</**Source-type**>, these results suggest that traditional techniques for measuring column densities – equivalent width, curve of growth, and Gaussian modeling – are of limited value when applied to UV absorption associated with <**Source-type**>active galactic nucleus</**Source-type**> outflows.

Figure 1. An example abstract.

Our choice of new domain was driven partly by longer-term plans to build an information extraction system for the astronomy domain and partly by the similarities and differences between this domain and the biomedical domain that the initial NER tagger is trained on. The main point of similarity between the two data sets is that they are both comprised of scientific language taken from abstracts of academic papers. The main difference lies in the technical terms and in the named entities that are recognised.

Following consultation with our astronomy collaborators, we created a cohesive dataset in the radio astronomy domain, and established an inventory of four domain-specific named entity types. The dataset was created by extracting abstracts from the years 1997-2003 that matched the query "quasar AND line". 50 abstracts from the year 2002 were annotated as seed material and 159 abstracts from 2003 were annotated as testing material. 778 abstracts from the years 1997-2001 were provided as an unannotated pool for bootstrapping. On average, these abstracts contain 10 sentences with an average length of 30 tokens. The corpus was annotated for the four entity types below (frequencies in the seed set in brackets). Fig. 1 shows an example text from this corpus.

Instrument-name Names of telescopes and other measurement instruments, e.g. *Superconducting Tunnel Junction (STJ) camera, Plateau de Bure Interferometer, Chandra, XMM-Newton Reflection Grating Spectrometer (RGS), Hubble Space Telescope.* [136 entities, 12.7%]

Source-name Names of celestial objects, e.g. *NGC 7603, 3C 273, BRI 1335-0417, SDSSp J104433.04-012502.2, PC0953+ 4749.* [111 entities, 10.4%]

Source-type Types of objects, e.g. *Type II Supernovae (SNe II), radio-loud quasar, type 2 QSO, starburst galaxies, low-luminosity AGNs.* [499 entities, 46.8%]

Spectral-feature Features that can be pointed to on a spectrum, e.g. *Mg II emission, broad emission lines, radio continuum emission at 1.47 GHz, CO ladder from (2-1) up to (7-6), non-LTE line.* [321 entities, 30.1%]

2.2. Corpus Preparation and Annotation

The files were converted from their original HTML to XHTML using Tidy (`http://www.w3.org/People/Raggett/tidy/`), and were piped through a sequence of processing stages using the XML-based tools from the LT TTT and LT XML toolsets (Grover et al., 2000; Thompson et al., 1997) in order to create tokenised XML files. It turned out to be relatively complex to achieve a sensible and consistent tokenisation of this data. The main source of complexity is the high density of technical and formulaic language (e.g. $(N(H_2) \simeq 10_{24} cm^{-2})$, 17.8 h_{70}^{-1} kpc, for $\Omega_m = 0.3$, $\Lambda = 0.7$, 1.4 GHz of 30 μ Jy) and an accompanying lack of consistency in the way publishers convert from the original LaTex encoding of formulae to the HTML which is published on the ADS website. We aimed to tokenise in such a way as to minimise noise in the data, though inevitably not all inconsistencies were removed.

The seed and test data sets were annotated by two astrophysics PhD students using the NITE XML toolkit annotation tool (Carletta et al., 2003). In addition, they annotated 1000 randomly sampled sentences from the pool to provide a baseline for active learning. Inter-annotator agreement was obtained by directly comparing the two annotator's data. Phrase-level f-score is 86.4%. Token-level accuracy is 97.3% which corresponds to a Kappa agreement of K=.925 (N=44775, k=2; where K is the kappa coefficient, N is the number of tokens and k is the number of annotators).

3. Active Learning with Multiple Views

Supervised training of named entity recognition (NER) systems requires large amounts of manually annotated data. However, human annotation is typically costly and time-consuming. Active learning promises to reduce this cost by requesting only those data points for human annotation which are highly informative. Example informativity can be estimated by the degree of uncertainty of a single learner as to the correct label of a data point (Cohn et al., 1995) or in terms of the disagreement of a committee of learners (Seung et al., 1992). Active learning has been successfully applied to a variety of similar tasks such as document classification (McCallum & Nigam, 1998), part-of-speech tagging (Argamon-Engelson & Dagan, 1999), and parsing (Thompson et al., 1999).

We employ a committee-based method where the degree of deviation of different classifiers with respect to their analysis can tell us if an example is potentially useful. Trained classifiers can be caused to be different by bagging (Abe & Mamitsuka, 1998), by randomly perturbing event counts (Argamon-Engelson & Dagan, 1999), or by producing different views using different feature sets for the same classifiers (Jones et al., 2003; Osborne & Baldridge, 2004). In this paper, we present active learning experiments for NER in astronomy texts following the last approach.

3.1. Feature split

We use a conditional Markov model tagger (Finkel et al., 2004) to train two different models on the same seed data by applying a feature split. The feature split as shown in Table 1 was hand-crafted such that it provides different views while empirically ensuring that performance is sufficiently similar. While the first feature set comprises of character sub-strings, BNC frequencies, Web counts, gazetteers and abbreviations, the second set contains features capturing information about words, POS tags, word shapes, NE tags, parentheses and multiple references to NEs. These features are describe in more detail in (Finkel et al., 2004).

Feature Set 1	
Prefix/Suffix	Up to a length of 6
Frequency	Frequency in BNC
Web Feature	Based on Google hits of pattern instantiations
Gazetteers	Compiled from the Web
Abbreviations	abbr_i
	abbr_i + abbr_{i+1}
	abbr_{i-1} + abbr_i + abbr_{i+1}
Feature Set 2	
Word Features	w_i, w_{i-1}, w_{i+1}
	Disjunction of 5 prev words
	Disjunction of 5 next words
TnT POS tags	POS_i, POS_{i-1}, POS_{i+1}
Word Shape	shape_i, shape_{i-1}, shape_{i+1}
	shape_i + shape_{i+1}
	shape_{i-1} + shape_i + shape_{i+1}
Prev NE	NE_{i-1}, NE_{i-2} + NE_{i-1}
	NE_{i-3} + NE_{i-2} + NE_{i-1}
Prev NE + Word	NE_{i-1} + w_i
Prev NE + POS	NE_{i-1} + POS_{i-1} + POS_i
	NE_{i-2} + NE_{i-1} + POS_{i-2} + POS_{i-1} + POS_i
Prev NE + Shape	NE_{i-1} + shape_i
	NE_{i-1} + shape_{i+1}
	NE_{i-1} + shape_{i-1} + shape_i
	NE_{i-2} + NE_{i-1} + shape_{i-2} + shape_{i-1} + shape_i
Paren-Matching	Signals when one parenthesis in a pair has been assigned a different tag in a window of 4 words
Occurrence Patterns	Capture multiple references to NEs

Table 1. Feature split for parameter optimisation experiments

3.2. Level of annotation

For the manual annotation of named entity examples, we needed to decide on the level of granularity. The question arises what constitutes an example that will be submitted to the annotators. Reasonable levels of annotation include the document level, the sentence level and the token level. The most fine-grained annotation would certainly be on the token level. This requires semi-supervised training to allow for partially annotated sentences, as in (Scheffer et al., 2001). However, there are no directly applicable semi-supervised training regimes for discriminative classifiers. On the other extreme, one may submit an entire document for annotation. A possible disadvantage is that a document with some interesting parts may well contain large portions with redundant, already known structures for which knowing the manual annotation may not be very useful. In the given setting, we decided that the best granularity is on the sentence level.

3.3. Sample Selection Metric

There are various metrics that could be used to quantify the degree of deviation between classifiers in a committee (e.g. KL-divergence, information radius, f-measure). The work reported here uses two sentence-level metrics based on KL-divergence and one based on f-score. In the following, we describe these metrics.

KL-divergence has been suggested for active learning to quantify the disagreement of classifiers over the probability distribution of output labels (McCallum & Nigam, 1998) and has been applied to information extraction (Jones et al., 2003). KL-divergence measures the divergence between two probability distributions p and q over the same event

space χ:

$$D(p||q) = \sum_{x \in \chi} p(x) \log \frac{p(x)}{q(x)} \quad (1)$$

KL-divergence is a non-negative metric. It is zero for identical distributions; the more different the two distributions, the higher the KL-divergence. KL-divergence is maximal for cases where distributions are peaked and prefer different labels. Taking a peaked distribution as an indicator for certainty, using KL-divergence thus bears a strong resemblance to the co-testing setting (Muslea, 2002). Intuitively, a high KL-divergence score indicates an informative data point. However, in the current formulation, KL-divergence only relates to individual tokens. In order to turn this into a sentence score, we need to combine the individual KL-divergences for the tokens within a sentence into one single score. We employed mean and max.

The *f-complement* has been suggested for active learning in the context of NP chunking as a structural comparison between the different analyses of a committee (Ngai & Yarowsky, 2000). It is the pairwise f-score comparison between the multiple analyses for a given sentence:

$$f^{\mathcal{M}}_{comp} = \frac{1}{2} \sum_{M,M' \in \mathcal{M}} (1 - F_1(M(t), M'(t))) \quad (2)$$

where F_1 is the balanced f-score of $M(t)$ and $M'(t)$, the preferred analyses of data point t according to different members M, M' of ensemble $\mathcal{M}$. The definition assumes that in the comparison between two analyses, one may arbitrarily assign one analysis as the gold standard and the other one as a test case. Intuitively, examples with a high f-complement score are likely to be informative.

4. Parameter Optimisation Experiments

In the previous section, we described a number of parameters for our approach to active learning. Bootstrapping presents a difficult problem as we cannot optimise these parameters on the target data. The obvious solution is to use a different data set but there is no guarantee that experimental results will generalise across domains. The work reported here addresses this question. We simulated active learning experiments on a data set which consists of biomedical abstracts marked up for the entities DNA, RNA, cell line, cell type, and protein (Kim et al., 2003).[1] Seed, pool, and test sets contained 500, 10,000, and 2,000 sentences respectively, roughly the same size as for the astronomical data. As smaller batch sizes require more retraining iterations and larger batch sizes increase the amount of annotation necessary at each round and could lead to unnecessary strain for the annotators, we settled on a batch size of 50 sentences for the real AL experiment as a compromise between computational cost and work load for the annotator.

We then ran simulated AL experiments for each of the three selection metrics discussed in section 3. The performance was compared to a baseline where examples were randomly sampled from the pool data. Experiments were run until there were 2000 sentences of annotated training material including the sentences from the seed data and the sentences selected from the pool data.

4.1. Costing Active Learning

For quality evaluation, we used the established f-score metric as given by the evaluation scripts developed for the CoNLL NER tasks (Tjong Kim Sang & De Meulder, 2003). In order to assess the relative merits of various active learning scenarios, we will plot learning curves, i.e. the increase in f-score over the invested effort. Ideally, a cost metric should reflect the effort that went into the annotation of examples in terms of time spent. However, a precise time measurement is difficult, or may be not available in the case of simulation experiments. We will therefore consider a number of possible approximations.

A sentence-based cost metric may seem like an obvious cost function, but this may pose problems when different sample selection metrics have a tendency to choose longer or shorter sentences. Thus, we will also consider more fine-grained metrics, namely the number of tokens in a sentence and the number of entities in a sentence.

4.2. Comparison of Selection Metrics

The plots in figure 2 show the learning curves for random sampling and the three AL selection metrics we examined for the parameter optimisation experiments. The first takes the number of sentences as the cost metric and the second and third take the number of tokens and the number of entities respectively.

Random sampling is clearly outperformed by all other selection metrics. The random curve for the sentence cost metric, for example, reaches an f-score of 69% after approximately 1500 sentences have been annotated while the maximum KL-divergence curve reaches this level of performance after only $\approx$ 1100 sentences. This represents a substantial reduction in sentences annotated of 26.7%. In addition, at 1500 sentences, maximum KL-divergence offers an error reduction of 4.9% over random sampling with a 1.5 point improvement in f-score. Averaged KL-divergence offers the same error reduction when using the sentence cost metric, but at 19.3%, a lower reduction of sentences annotated. F-complement performs worst giving 10% cost reduction and 1.6% error reduction.

The learning curves also allow us to easily visualise the

[1] Simulated AL experiments use 5-fold cross-validation.

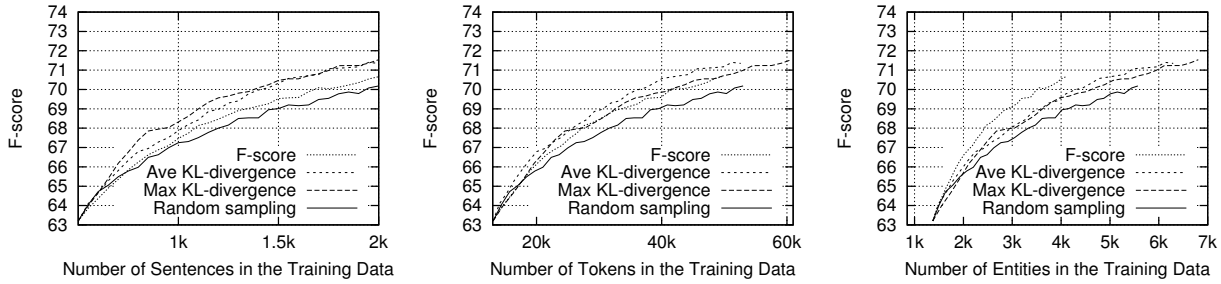


Figure 2. Parameter optimisation learning curves for sentence, token, and entity cost metrics

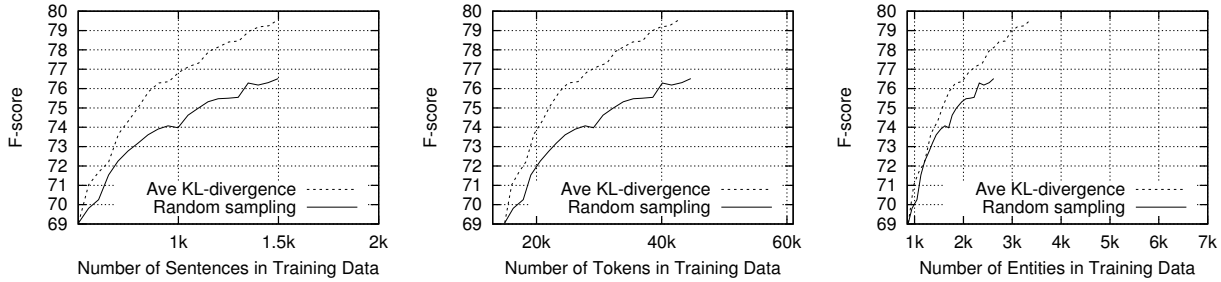


Figure 3. Active annotation learning curves for sentence, token, and entity cost metrics

performance difference of the three selection metrics with respect to each other. The f-complement metric clearly underperforms with respect to KL-divergence based metrics.

According to the learning curves with number of sentences as the cost metric, maximum KL-divergence performs the best. However, when choosing a different cost metric, as for example the number of tokens or entities that occur in each selected sentence, the learning curves behave completely differently as can be seen in the second and third plots in figure 2. This illustrates the fact that the selection metrics operate in different ways preferring shorter or longer sentences with more or less entities. With number of tokens as the cost metric, averaged KL-divergence performs the best with a 23.5% reduction in annotation cost to reach an f-score of 69% and an error reduction of 4.9% at $\approx$ 40,000 tokens. And with entities as the cost metric, the f-complement selection metric seems to perform best. So, the question arises: how do we combine this information to prepare for a real annotation task where we only have a single opportunity to get the best performing and most cost effective system possible.

To explore the behaviour of the three selection metrics further, we also look at the number of tokens and the number of entities in the sentences chosen by each metric. Table 2 contains the number of tokens and entities contained within the selected sentences averaged across the 5 cross-validation results. Comparing these numbers, one can observe the types of sentences preferred by each selection

Metric	Tokens	Entities
Random	26.7 (0.8)	2.8 (0.1)
F-comp	25.8 (2.4)	2.2 (0.7)
KL-max	30.9 (1.5)	3.5 (0.2)
KL-ave	27.1 (1.8)	3.3 (0.2)

Table 2. Average tokens and entities per sentence for different selection metrics (standard deviation in brackets)

metric. While the maximum KL-divergence metric selects the longest sentences containing the most number of entities, the f-complement selection metric chooses the shortest sentences with the least number of entities in them. The averaged KL-divergence metric, on the other hand, generally selects average length sentences which still contain relatively many entities.

As averaged KL-divergence does not affect sentence length, we expect the sentences selected to take less time to annotate than the sentences selected by maximum KL-divergence. And, since these sentences have relatively many entity phrases, we expect to have more positive examples than with the f-complement metric and thus have higher informativity and therefore performance increase per token. Furthermore, sentence length is not the best single unit cost metric. The number of sentences is too coarse as this gives the same cost to very long and very short sentences and does not allow us to consider the types of sentences selected by the various metrics. Likewise, the number of entities does not reflect the fact that every selected

sentence needs to be read regardless of the number of entities it contains, which again covers up effects of specific selection metrics.

5. Active Annotation Results

We developed NEAL, an interactive Named Entity Active Learning tool for bootstrapping NER in a new domain. The tool manages the data and presents batches of selectively sampled sentences for annotation in the same annotation tool used for the seed and test data. The entire abstract is presented for context with the target sentence(s) highlighted. On the basis of the findings of the simulated experiments we set up the real AL experiment using averaged KL-divergence as the selection metric. The tool was initialised with the 50 document seed set described in section 2 and given to the same annotators that prepared the seed and test sets.

As we do not have a model of temporal or monetary cost in terms of our three cost metrics, we evaluate with respect to all three cost metrics. Figure 3 contains learning curves for random sampling and for selective sampling with the averaged KL-divergence selection metric plotted against number of sentences, number of tokens, and number of entities. The initial performance (given only the seed data for training) amounts to an f-score of 69.1%. 50 sentences (with an average of 28 tokens and 2.5 entities per sentence as compared to 29.8 and 2.0 for the randomly sampled data) are added to the training data at each round. After 20 iterations, the training data therefore comprises of 1,502 sentences (containing approx. 43,000 tokens) which leads to an f-score of 79.6%.

Comparing the selective sampling performance to the baseline, we confirm that active learning provides a significant reduction in the number of examples that need annotating. Looking first at the token cost metric, the random curve reaches an f-score of 76% after approximately 39,000 tokens of data has been annotated while the selective sampling curve reaches this level of performance after only $\approx$ 24,000 tokens. As for the optimisation data, this represents a dramatic reduction in tokens annotated of 38.5%. In addition, at 39,000 tokens, selectively sampling offers an error reduction of 13.0% with a 3 point improvement in f-score. Selective sampling with the averaged KL-divergence selection metric also achieves dramatic cost and error rate reductions for the sentence (35.6% & 12.5%) and entity cost metrics (23.9% & 5.0%).

These improvements are comparable to the cost and error reduction achieved in the optimisation data. While it should be taken into account that these domains are relatively similar, this suggests that a different domain can be used to optimise parameters when using active learning to bootstrap NER. This is confirmed not only by an improvement over baseline for the token cost metric but also by an improvement for the sentence and entity cost metrics.

In a companion paper, we report in some more detail about the effects of selective sampling on annotator's performance (Hachey et al., 2005). Even though we find that active learning may result in a slightly higher error rate in the annotation, we demonstrate that active learning still incurs substantial reductions in annotation effort as compared to random sampling.

6. Conclusions and Future Work

We have presented an active learning approach to bootstrapping named entity recognition for which a new corpus of radio astronomical texts has been collected and annotated. We employ a committee-based method that uses two different feature sets for a conditional Markov model tagger and we experiment with several metrics for quantifying the degree of deviation: averaged KL-divergence, maximum KL-divergence, and f-complement.

We started with a NER system tested and optimised in a domain with existing corpus resources and built a system to identify four novel entity types in a new domain of astronomy texts. Experimental results from the real active learning annotation illustrate that the optimised parameters performed well on the new domain. This is confirmed for cost metrics based on the number of sentences, the number of tokens, and the number of entities.

While presenting results with respect to the three cost metrics ensures completeness, it also suggests that the real cost might be better modelled as a combination of these metrics. During annotation, we collected timing information for each sentence and we are currently using this timing information to investigate realistic models of cost based on sentence length and number of entities.

Acknowledgments

We are very grateful for the time and resources invested in corpus preparation by our collaborators in the Institute for Astronomy, University of Edinburgh: Rachel Dowsett, Olivia Johnson and Bob Mann. We would also like to thank Shipra Dingare, Jochen Leidner, Malvina Nissim and Yuval Krymolowski for helpful discussions. Many thanks to Ewan Klein, Miles Osborne, and Bonnie Webber for being instrumental in formulating and organising the task.

We are grateful to the UK National e-Science Centre for giving us access to BlueDwarf, a p690 server donated to the University of Edinburgh by IBM. This work was performed as part of the SEER project, which is supported by a Scottish Enterprise Edinburgh-Stanford Link Grant (R36759).

References

Abe, N., & Mamitsuka, H. (1998). Query learning strategies using boosting and bagging. *Proceedings of the 15th International Conference on Machine Learning*.

Argamon-Engelson, S., & Dagan, I. (1999). Committee-based sample selection for probabilistic classifiers. *Journal of Artificial Intelligence Research*, *11*, 335–360.

Carletta, J., Evert, S., Heid, U., Kilgour, J., Robertson, J., & Voormann, H. (2003). The NITE XML Toolkit: flexible annotation for multi-modal language data. *Behavior Research Methods, Instruments, and Computers, special issue on Measuring Behavior*, *35*.

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1995). Active learning with statistical models. *Advances in Neural Information Processing Systems* (pp. 705–712). The MIT Press.

Finkel, J., Dingare, S., Nguyen, H., Nissim, M., Manning, C., & Sinclair, G. (2004). Exploiting context for biomedical entity recognition: From syntax to the web. *Proceedings of the COLING 2004 International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*.

Grover, C., Matheson, C., Mikheev, A., & Moens, M. (2000). LT TTT—a flexible tokenisation tool. *Proceedings of the 2nd International Conference on Language Resources and Evaluation*.

Hachey, B., Alex, B., & Becker, M. (2005). Investigating the effects of selective sampling on the annotation task. *Proceedings of CoNLL 2005, Ann Arbor, USA*.

Jones, R., Ghani, R., Mitchell, T., & Riloff, E. (2003). Active learning for information extraction with multiple view feature sets. *ECML 2003 Workshop on Adaptive Text Extraction and Mining*.

Kim, J.-D., Ohta, T., Tateishi, Y., & Tsujii, J. (2003). Genia corpus - a semantically annotated corpus for bio-textmining. *Bioinformatics*, *19*, 180–182.

McCallum, A., & Nigam, K. (1998). Employing EM and pool-based active learning for text classification. *Proceedings of the 15th International Conference on Machine Learning*.

Muslea, I. (2002). *Active learning with multiple views*. Doctoral dissertation, University of Southern California.

Ngai, G., & Yarowsky, D. (2000). Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*.

Osborne, M., & Baldridge, J. (2004). Ensemble-based active learning for parse selection. *Proceedings of the 5th Conference of the North American Chapter of the Association for Computational Linguistics*.

Scheffer, T., Decomain, C., & Wrobel, S. (2001). Active hidden markov models for information extraction. *Proceedings of the International Symposium on Intelligent Data Analysis*.

Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*.

Thompson, C. A., Califf, M. E., & Mooney, R. J. (1999). Active learning for natural language parsing and information extraction. *Proceedings of the 16th International Conference on Machine Learning*.

Thompson, H., Tobin, R., McKelvie, D., & Brew, C. (1997). LT XML. Software API and toolkit for XML processing. `http://www.ltg.ed.ac.uk/software/`.

Tjong Kim Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *Proceedings of the 2003 Conference on Computational Natural Language Learning*.

Estimation of Mixture Models using Co-EM

Steffen Bickel BICKEL@INFORMATIK.HU-BERLIN.DE
Tobias Scheffer SCHEFFER@INFORMATIK.HU-BERLIN.DE
Humboldt-Universität zu Berlin, Department of Computer Science, Unter den Linden 6, 10099 Berlin, Germany

Abstract

We study estimation of mixture models for problems in which multiple views of the instances are available. Examples of this setting include clustering web pages or research papers that have intrinsic (text) and extrinsic (references) attributes. Our optimization criterion quantifies the likelihood and the consensus among models in the individual views; maximizing this consensus minimizes a bound on the risk of assigning an instance to an incorrect mixture component. We derive an algorithm that maximizes this criterion. Empirically, we observe that the resulting clustering method incurs a lower cluster entropy than regular EM for web pages, research papers, and many text collections.

1. Introduction

In many application domains, instances can be represented in two or more distinct, redundant views. For instance, web pages can be represented by their text, or by the anchor text of inbound hyperlinks ("miserable failure"), and research papers can be represented by their references from and to other papers, in addition to their content. In this case, multi-view methods such as co-training (Blum & Mitchell, 1998) can learn two initially independent hypotheses. These hypotheses bootstrap by providing each other with conjectured class labels for unlabeled data. Multi-view learning has often proven to utilize unlabeled data effectively, increase the accuracy of classifiers (*e.g.,* Yarowsky, 1995; Blum & Mitchell, 1998) and improve the quality of clusterings (Bickel & Scheffer, 2004).

Nigam and Ghani (2000) propose the co-EM procedure that resembles semi-supervised learning with EM (McCallum & Nigam, 1998), using two views that alternate after each iteration. The EM algorithm (Dempster et al., 1977) is very well understood. In each iteration, it maximizes the expected joint log-likelihood of visible and invisible data given the parameter estimates of the previous iteration — the Q function. This procedure is known to greedily maximize the likelihood of the data. By contrast, the primary justification of the co-EM algorithm is that it often works very well; it is not known which criterion the method maximizes.

We take a top down approach on the problem of mixture model estimation in a multi-view setting. A result of Dasgupta et al. (2001) motivates our work by showing that a high consensus of independent hypotheses implies a low error rate. We construct a criterion that quantifies likelihood and consensus and derive a procedure that maximizes it. We contribute to an understanding of mixture model estimation for multiple views by showing that the co-EM algorithm is a special case of the resulting procedure. Our solution naturally generalizes co-EM for more than two views. We show that a variant of the method in which the consensus term is annealed over time is guaranteed to converge.

The rest of this paper is organized as follows. Section 2 discusses related work. In Section 3, we define the problem setting. Section 4 motivates our approach, discusses the new Q function, the unsupervised co-EM algorithm, and its instantiation for mixture of multinomials. We conduct experiments in Section 5 and conclude with Section 6.

2. Related Work

Most studies on multi-view learning address semi-supervised classification problems. de Sa (1994) observes a relationship between consensus of multiple hypotheses and their error rate and devised a semi-supervised learning method by cascading multi-view vector quantization and linear classification. A multi-view approach to word sense disambiguation combines a classifier that refers to the local context of a word with a second classifier that utilizes the document in which words co-occur (Yarowsky, 1995). Blum and

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

Mitchell (1998) introduce the co-training algorithm for semi-supervised learning that greedily augments the training set of two classifiers. A version of the AdaBoost algorithm boosts the agreement between two views on unlabeled data (Collins & Singer, 1999).

Dasgupta et al. (2001) and Abney (2002) give PAC bounds on the error of co-training in terms of the disagreement rate of hypotheses on unlabeled data in two independent views. This justifies the direct minimization of the disagreement. The co-EM algorithm for semi-supervised learning probabilistically labels all unlabeled examples and iteratively exchanges those labels between two views (Nigam & Ghani, 2000; Ghani, 2002). Muslea et al. (2002) extend co-EM for active learning. Brefeld and Scheffer (2004) study a co-EM wrapper for the Support Vector Machine.

For unsupervised learning, several methods combine models that are learned using distinct attribute subsets in a way that encourages agreement. Becker and Hinton (1992) maximize mutual information between the output of neural network modules that perceive distinct views of the data. Models of images and their textual annotations have been combined (Barnard et al., 2002; Blei & Jordan, 2003). Reinforcement clustering (Wang et al., 2003) exchanges cluster membership information between views by artificial attributes. Bickel and Scheffer (2004) use the co-EM algorithm for clustering of data with two views. Clustering by maximizing the dependency between views is studied by Sinkkonen et al. (2004). Also, the density-based DBSCAN clustering algorithm has a multi-view counterpart (Kailing et al., 2004).

3. Problem Setting

The *multi-view* setting is characterized by available attributes X which are decomposed into views $X^{(1)}, \ldots, X^{(s)}$. An instance $x = (x^{(1)}, \ldots, x^{(s)})$ has representations $x^{(v)}$ that are vectors over $X^{(v)}$. We focus on the problem of estimating parameters of a generative mixture model in which data are generated as follows. The *data generation process* selects a mixture component j with probability α_j. Mixture component j is the value of a random variable Z. Once j is fixed, the generation process draws the s independent vectors $x^{(v)}$ according to the likelihoods $P(x^{(v)}|j)$. The likelihoods $P(x^{(v)}|j)$ are assumed to follow a parametric model $P(x^{(v)}|j, \Theta)$ (distinct views may of course be governed by distinct distributional models).

The *learning task* involved is to estimate the parameters $\Theta = (\Theta^{(1)}, \ldots, \Theta^{(s)})$ from data. The *sample* consists of n observations that usually contain only the visible attributes $x_i^{(v)}$ in all views v of the instances x_i. The vector Θ contains priors $\alpha_j^{(v)}$ and parameters of the likelihood $P(x_i^{(v)}|j, \Theta^{(v)})$, where $1 \leq j \leq m$ and m is the number of mixture components assumed by the model (clusters). Given Θ, we will be able to calculate a posterior $P(j|x^{(1)}, \ldots, x^{(s)}, \Theta)$. This posterior will allow us to assign a cluster membership to any instance $x = (x^{(1)}, \ldots, x^{(s)})$. The *evaluation metric* is the impurity of the clusters as measured by the entropy; the elements of each identified cluster should originate from the same true mixture component.

4. Derivation of the Algorithm

Dasgupta et al. (2001) have studied the relation between the consensus among two independent hypotheses and their error rate. Let us review a very simple result that motivates our approach, it can easily be derived from their general treatment of the topic. Let $h^{(v)}(x) = \text{argmax}_j \; P(j|x^{(v)}, \Theta^{(v)})$ be two independent clustering hypotheses in views $v = 1, 2$. For clarity of the presentation, let there be two true mixture components. Let x be a randomly drawn instance that, without loss of generality belongs to mixture component 1, and let both hypotheses $h^{(1)}$ and $h^{(2)}$ have a probability of at least 50% of assigning x to the correct cluster 1. We observe that

$$P(h^{(1)}(x) \neq h^{(2)}(x)) \geq \max_v P(h^{(v)}(x) \neq 1).$$

That is, the probability of a disagreement $h^{(1)}(x) \neq h^{(2)}(x)$ is an upper bound on the risk of an error $P(h^{(v)}(x) \neq 1)$ of either hypothesis $h^{(v)}$.

We give a brief *proof* of this observation. In Equation 1 we distinguish between the two possible cases of disagreement; we utilize the independence assumption and order the summands such that the greater one comes first. In Equation 2, we exploit that the error rate be at most 50%: both hypotheses are less likely to be wrong than just one of them. Exploiting the independence again takes us to Equation 3.

$$\begin{aligned}
& P(h^{(1)}(x) \neq h^{(2)}(x)) \\
&= P(h^{(v)}(x) = 1, h^{(\bar{v})}(x) = 2) + \\
&\quad P(h^{(v)}(x) = 2, h^{(\bar{v})}(x) = 1) && (1) \\
&\quad \text{where } v = \text{argmax}_u P(h^{(u)}(x) = 1, h^{(\bar{u})}(x) = 2) \\
&\geq P(h^{(v)}(x) = 2, h^{(\bar{v})}(x) = 2) + \\
&\quad P(h^{(v)}(x) = 2, h^{(\bar{v})}(x) = 1) && (2) \\
&= \max_v P(h^{(v)}(x) \neq 1) && (3)
\end{aligned}$$

In unsupervised learning, the risk of assigning instances to wrong mixture components cannot be minimized directly, but with the above argument we can minimize an upper bound on this risk.

The Q function is the core of the EM algorithm. We will now review the usual definition, include a consensus term, and find a maximization procedure.

4.1. Single-View Optimization Criterion

Even though the goal is to maximize $P(X|\Theta)$, EM iteratively maximizes an auxiliary (single-view) criterion $Q^{SV}(\Theta, \Theta_t)$. The criterion refers to the visible variables X, the invisibles Z (the mixture component), the optimization parameter Θ and the parameter estimates Θ_t of the last iteration. Equation 4 defines $Q^{SV}(\Theta, \Theta_t)$ to be the expected log-likelihood of $P(X, Z|\Theta)$, given X and given that the hidden mixture component Z be distributed according to $P(j|x, \Theta_t)$.

The criterion $Q^{SV}(\Theta, \Theta_t)$ can be determined as in Equation 5 for mixture models. It requires calculation of the posterior $P(j|x_i, \Theta_t)$ as in Equation 6; this is referred to as the E step of the EM algorithm. In the M step, it finds the new parameters $\Theta_{t+1} = \text{argmax}_\Theta\, Q^{SV}(\Theta, \Theta_t)$ that maximize Q^{SV} over Θ. The parameters Θ occur in Equation 5 only in the prior probabilities α_j and likelihood terms $P(x_i|j, \Theta)$.

$$Q^{SV}(\Theta, \Theta_t) = E[\log P(X, Z|\Theta)|X, \Theta_t] \tag{4}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i, \Theta_t) \log(\alpha_j P(x_i|j, \Theta)) \tag{5}$$

$$P(j|x_i, \Theta_t) = \frac{\alpha_j P(x_i|j, \Theta_t)}{\sum_k \alpha_k P(x_i|k, \Theta_t)} \tag{6}$$

The EM algorithm starts with some initial guess at the parameters Θ_0 and alternates E and M steps until convergence. Dempster et al. (1977) prove that, in each iteration, $P(X|\Theta_{t+1}) - P(X|\Theta_t) \geq 0$. Wu (1983) furthermore proves conditions for the convergence of the sequence of parameters $(\Theta)_t$.

4.2. Multi-View Criterion

We want to maximize the likelihood in the individual views and the consensus of the models because we know that the disagreement bounds the risk of assigning an instance to an incorrect mixture component. Equations 7 and 8 define our *multi-view Q function* as the sum over s single-view Q functions minus a penalty term $\Delta(\cdot)$ that quantifies the disagreement of the models $\Theta^{(v)}$ and is regularized by η.

$$Q^{MV}(\Theta^{(1)}, \ldots, \Theta^{(s)}, \Theta_t^{(1)}, \ldots, \Theta_t^{(s)}) = \sum_{v=1}^{s} Q^{SV}(\Theta^{(v)}, \Theta_t^{(v)}) - \eta\Delta(\Theta^{(1)}, \ldots, \Theta^{(s)}, \Theta_t^{(1)}, \ldots, \Theta_t^{(s)}) \tag{7}$$

$$= \sum_{v=1}^{s} E\left[\log P(X^{(v)}, Z^{(v)}|\Theta^{(v)})|X^{(v)}, \Theta_t^{(v)}\right] - \eta\Delta(\Theta^{(1)}, \ldots, \Theta^{(s)}, \Theta_t^{(1)}, \ldots, \Theta_t^{(s)}) \tag{8}$$

When the regularization parameter η is zero, then $Q^{MV} = \sum_v Q^{SV}$. In each step, co-EM then maximizes the s terms Q^{SV} independently. It follows immediately from Dempster et al. (1977) that each $P(X^{(v)}|\Theta^{(v)})$ increases in each step and therefore $\sum_v P(X^{(v)}|\Theta^{(v)})$ is maximized.

The disagreement term Δ should satisfy a number of desiderata. Firstly, since we want to minimize Δ, it should be convex. Secondly, for the same reason, it should be differentiable. Given Θ_t, we would like to find the maximum of $Q^{MV}(\Theta, \Theta_t)$ in one single step. We would, thirdly, appreciate if Δ was zero when the views totally agree.

We construct Δ to fulfill these desiderata in Equation 9. It contains the pairwise cross entropy $H(P(j|x_i^{(v)}, \Theta_t^{(v)}), P(j|x_i^{(u)}, \Theta^{(u)}))$ of the posteriors of any pair of views u and v. The second cross entropy term $H(P(j|x_i^{(v)}, \Theta_t^{(v)}), P(j|x_i^{(v)}, \Theta^{(v)}))$ scales Δ down to zero when the views totally agree. Equation 10 expands all cross-entropy terms. At an abstract level, Δ can be thought of as all pairwise Kullback-Leibler divergences of the posteriors between all views. Since the cross entropy is convex, Δ is convex, too.

$$\Delta(\Theta^{(1)}, \ldots, \Theta^{(s)}, \Theta_t^{(1)}, \ldots, \Theta_t^{(s)}) = \frac{1}{s-1}\sum_{v\neq u}\sum_{i=1}^{n}\Big(H(P(j|x_i^{(v)}, \Theta_t^{(v)}), P(j|x_i^{(u)}, \Theta^{(u)})) - H(P(j|x_i^{(v)}, \Theta_t^{(v)}), P(j|x_i^{(v)}, \Theta^{(v)}))\Big) \tag{9}$$

$$= \frac{1}{s-1}\sum_{v\neq u}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)}, \Theta_t^{(v)}) \log\frac{P(j|x_i^{(v)}, \Theta^{(v)})}{P(j|x_i^{(u)}, \Theta^{(u)})} \tag{10}$$

In order to implement the M step, we have to maximize $Q^{MV}(\Theta, \Theta_t)$ given Θ_t. We have to set the derivative to zero. Parameter Θ occurs in the logarithmized posteriors, so we have to differentiate a sum of likelihoods within a logarithm. Theorem 1 solves this problem and rewrites Q^{MV} analogously to Equation 5.

Equation 12 paves the way to an algorithm that maximizes Q^{MV}. The parameters Θ occur only in the log-likelihood terms $\log P(x_i^{(v)}|j, \Theta^{(v)})$ and $\log \alpha_j^{(v)}$ terms, and Q^{MV} can be rewritten as a sum over local functions Q_v^{MV} for the views v. It now becomes clear that the M step can be executed by finding parameter estimates of $P(x_i^{(v)}|j, \Theta^{(v)})$ and $\alpha_j^{(v)}$ independently in each view v. The E step can be carried out by calculating and averaging the posteriors $P^{(v)}(j|x_i, \Theta_t, \eta)$ according to Equation 13; this equation specifies how the views interact.

Theorem 1 *The multi-view criterion Q can be expressed as a sum of local functions Q_v^{MV} (Equation 11) that can be maximized independently in each view v. The criterion can be calculated as in Equation 12, where $P^{(v)}(j|x_i, \Theta_t, \eta)$ is the averaged posterior as detailed in Equation 13 and $P(j|x_i^{(v)}, \Theta_t^{(v)})$ is the local posterior of view v, detailed in Equation 14.*

$$Q^{MV}(\Theta^{(1)}, \ldots, \Theta^{(s)}, \Theta_t^{(1)}, \ldots, \Theta_t^{(s)})$$

$$= \sum_{v=1}^{s} Q_v^{MV}(\Theta^{(v)}, \Theta_t^{(1)}, \ldots, \Theta_t^{(s)}) \quad (11)$$

$$= \sum_{v=1}^{s} \left(\sum_{i=1}^{n} \sum_{j=1}^{m} P^{(v)}(j|x_i, \Theta_t, \eta) \log \alpha_j^{(v)} \right. \quad (12)$$

$$\left. + \sum_{i=1}^{n} \sum_{j=1}^{m} P^{(v)}(j|x_i, \Theta_t, \eta) \log P(x_i^{(v)}|j, \Theta^{(v)}) \right)$$

$$P^{(v)}(j|x_i, \Theta_t^{(1)}, \ldots, \Theta_t^{(s)}, \eta) \quad (13)$$

$$= (1-\eta) P(j|x_i^{(v)}, \Theta_t^{(v)}) + \frac{\eta}{s-1} \sum_{\bar{v} \neq v} P(j|x_i^{(\bar{v})}, \Theta_t^{(\bar{v})})$$

$$P(j|x_i^{(v)}, \Theta_t^{(v)}) = \frac{\alpha_j^{(v)} P(x_i^{(v)}|j, \Theta_t^{(v)})}{\sum_k \alpha_k^{(v)} P(x_i^{(v)}|k, \Theta_t^{(v)})} \quad (14)$$

The proof for Theorem 1 is given in Appendix A.

4.3. Generalized Co-EM Algorithm

Theorem 1 describes the unsupervised co-EM algorithm with arbitrarily many views mathematically. The M steps can be executed independently in the views but Theorem 1 leaves open how the E and M steps should be interleaved. Co-EM can be implemented such that a global E step is followed by M steps in all views or, alternatively, we can iterate over the views in an outer loop and execute an E and an M step in the current view in each iteration of this loop.

We implement the latter strategy because consecutive M steps in multiple views impose the following risk. Cases can arise in which Q_1^{MV} can be maximized by changing $\Theta_{t+1}^{(1)}$ such that it agrees with $\Theta_t^{(2)}$. A consecutive M step in view 2 can then maximize Q_2^{MV} by changing $\Theta_{t+1}^{(2)}$ such that it agrees with $\Theta_t^{(1)}$. As a result, the two models flip their dissenting opinions. We observe empirically that this effect slows down the convergence; if the Q function consisted of only the Δ term, then this could even lead to alternation.

The unsupervised co-EM algorithm with multiple views is shown in Table 1. When the execution has reached time step t and view v, the parameters $\Theta_{t+1}^{(1)}, \ldots, \Theta_{t+1}^{(v-1)}$ and $\Theta_t^{(v)}, \ldots, \Theta_t^{(s)}$ have already been estimated. In the E step, we can therefore determine the posterior $P^{(v)}(j|x_i, \Theta_{t+1}^{(1)}, \ldots, \Theta_{t+1}^{(v-1)}, \Theta_t^{(v)}, \ldots, \Theta_t^{(s)}, \eta)$ using the most recent parameter estimates. In the succeeding M step, the local Q_v^{MV} function is maximized over the parameter $\Theta^{(v)}$. Note that the co-EM algorithm of Nigam and Ghani (2000) is a special case of Table 1 for two views, $\eta = 1$, and semi-supervised instead of unsupervised learning.

Table 1. Unsupervised Co-EM Algorithm with Multiple Views.

Input: Unlabeled data $(x_i^{(1)}, \ldots, x_i^{(s)}) \in D$. Regularization parameter η (by default, 1).

1. Initialize $\Theta_0^{(1)}, \ldots, \Theta_0^{(s)}$ at random; let $t = 1$.
2. Do until convergence of Q^{MV}:
   (a) For $v = 1 \ldots s$:
      i. E step in view v: Compute the posterior $P^{(v)}(j|x_i, \Theta_{t+1}^{(1)}, \ldots, \Theta_{t+1}^{(v-1)}, \Theta_t^{(v)}, \ldots, \Theta_t^{(s)}, \eta)$ in view v using Equation 13.
      ii. M step in view v: maximize Q^{MV}; $\Theta_{t+1}^{(v)} = \text{argmax}_{\Theta^{(v)}} Q_v^{MV}(\Theta^{(v)}, \Theta_{t+1}^{(1)}, \ldots, \Theta_{t+1}^{(v-1)}, \Theta_t^{(v)}, \ldots, \Theta_t^{(s)})$.
   (c) Increment t.
3. Return $\Theta = (\Theta_t^{(1)}, \ldots, \Theta_t^{(s)})$.

In every step 2(a)ii, the local function Q_v^{MV} increases. Since all other $Q_{\bar{v}}^{MV}$ are constant in $\Theta^{(v)}$, this implies that also the global function Q^{MV} increases. In each iteration of the regular EM algorithm, $P(X|\Theta_{t+1}) - P(X|\Theta_t) \geq 0$. For co-EM, this is clearly not the case since the Q function has been augmented by a dissent penalization term. Wu (1983) proves conditions for the convergence of the sequence $(\Theta)_t$ for regular EM. Sadly, the proof does not transfer to co-EM.

We study a variant of the algorithm for which convergence can be proven. In an additional step 2(b), η is decremented towards zero according to some annealing scheme. This method can be guaranteed to converge; the proof is easily derived from the convergence guarantees of regular EM (Dempster et al., 1977; Wu, 1983). We can furthermore show that co-EM with annealing of η maximizes $\sum_v P(X^{(v)}|\Theta)$. In the beginning of the optimization process, Δ contributes strongly to the criterion Q^{MV}; the dissent Δ is convex and we know that it upper-bounds the error. Therefore, Δ guides the search to a parameter region of low error. The contribution of Δ vanishes later; $\sum_v P(X^{(v)}|\Theta)$ usually has many local maxima and having added Δ earlier now serves as a heuristic that may lead to a good local maximum.

4.4. Global Prior Probabilities

According to our generative model we have one global prior for each mixture component, but in step 2(a)ii the co-EM algorithm so far estimates priors in each view v from the data. We will now focus on maximization of Q subject to the constraint that the estimated priors of all views be equal.

We introduce two sets of Lagrange multipliers and get Lagrangian $L(\alpha, \lambda, \gamma)$ in Equation 15. Multiplier $\lambda^{(v)}$ guarantees that $\sum_j \alpha_j^{(v)} = 1$ in view v and $\gamma^{(j,v)}$ enforces the constraint $\alpha_j^{(1)} = \alpha_j^{(v)}$ for component j.

$$L(\alpha, \lambda, \gamma) = \sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m} P^{(v)}(j|x_i, \Theta_t, \eta) \log \alpha_j^{(v)} + \sum_{v=1}^{s} \lambda^{(v)} \left(\sum_{j=1}^{m} \alpha_j^{(v)} - 1 \right) + \sum_{v=2}^{s}\sum_{j=1}^{m} \gamma^{(j,v)}(\alpha_j^{(1)} - \alpha_j^{(v)}) \quad (15)$$

Setting the partial derivatives of $L(\alpha, \lambda, \gamma)$ to zero and solving the resulting system of equations leads to Equation 16. Expanding $P^{(v)}(j|x_i, \Theta_t, \eta)$, the regularization parameter η cancels out and we reach the final M step for $\alpha_j^{(v)}$ in Equation 17. We can see that the estimated prior is an average over all views and is therefore equal for all views.

$$\alpha_j^{(v)} = \frac{1}{sn}\sum_{v=1}^{s}\sum_{i=1}^{n} P^{(v)}(j|x_i, \Theta_t, \eta) \quad (16)$$

$$= \frac{1}{sn}\sum_{v=1}^{s}\sum_{i=1}^{n} P(j|x_i^{(v)}, \Theta_t^{(v)}) = \alpha_j \quad (17)$$

4.5. Cluster Assignment

For cluster analysis, an assignment of instances to clusters has to be derived from the model parameters. The risk of deciding for an incorrect cluster is minimized by choosing the *maximum a posteriori* hypothesis as in Equation 18. Bayes' rule and the conditional independence assumption lead to Equation 19.

$$h(x_i) = \text{argmax}_j P(j|x_i, \Theta) \quad (18)$$

$$= \underset{j}{\text{argmax}} \frac{\alpha_j \prod_{v=1}^{s} P(x_i^{(v)}|j, \Theta^{(v)})}{\sum_k^m \alpha_k \prod_{v=1}^{s} P(x_i^{(v)}|k, \Theta^{(v)})} \quad (19)$$

4.6. Mixture of Multinomials

In step 2(a)ii the co-EM algorithm estimates parameters in view v from the data. This step is instantiated for the specific distributional model used in a given application. We will detail the maximization steps for multinomial models which we use in our experimentation because they model both text and link data appropriately.

A multinomial model j is parameterized by the probabilities $\theta_{lj}^{(v)}$ of word w_l in view v and mixture component j. The likelihood of document $x_i^{(v)}$ is given by Equation 20. Parameters $n_{il}^{(v)}$ count the occurrences of word w_l in document $x_i^{(v)}$. $P(|x_i^{(v)}|)$ is the prior on the document length. The factorials account for all possible sequences that result in the set of words $x_i^{(v)}$.

$$P(x_i^{(v)}|j, \Theta^{(v)}) = P(|x_i^{(v)}|)|x_i^{(v)}|! \prod_l \frac{(\theta_{lj}^{(v)})^{n_{il}^{(v)}}}{n_{il}^{(v)}!} \quad (20)$$

We will now focus on maximization of Q^{MV} over the parameters $\theta_{lj}^{(v)}$. Lagrangian $L(\theta, \lambda)$ in Equation 21 guarantees that the word probabilities sum to one.

$$L(\theta, \lambda) = \sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m} P^{(v)}(j|x_i, \Theta_t, \eta) \left(\log P(|x_i^{(v)}|)|x_i^{(v)}|! + \sum_l n_{il}^{(v)} (\log \frac{\theta_{lj}^{(v)}}{n_{il}^{(v)}!} \right) + \sum_{v=1}^{s}\sum_{j=1}^{m} \lambda_j^{(v)} \left(\sum_l \theta_{lj}^{(v)} - 1 \right) \quad (21)$$

Setting the partial derivatives to zero and solving the resulting system of equations yields Equation 22.

$$\theta_{lj}^{(v)} = \frac{\sum_i P^{(v)}(j|x_i, \Theta_t, \eta) n_{il}^{(v)}}{\sum_k \sum_i P^{(v)}(j|x_i, \Theta_t, \eta) n_{ik}^{(v)}} \quad (22)$$

5. Empirical Studies

We want to find out (1) whether co-EM with multiple views finds better clusters in sets of linked documents with mixture of multinomials than regular single-view EM; (2) whether co-EM is still beneficial when there is no natural feature split in the data; (3) whether there are problems for which the optimal number of views lies above 2; and (4) whether the consensus regularization parameter η should be annealed or fixed to some value. To answer these questions, we experiment on archives of linked and plain text documents. All data sets that we use contain labeled instances; the labels are not visible to the learning method but we use them to measure the impurity of the returned clusters. Our quality measure is the average entropy over all clusters (Equation 23). This measure corresponds to the average number of bits needed to code the real class labels given the clustering result. The frequency $\hat{p}_{i|j}$ counts the number of elements of class i in cluster j, and n_j is the size of cluster j.

$$H = \sum_{j=1}^{m} \frac{n_j}{n} \left(-\sum_i \hat{p}_{i|j} \log \hat{p}_{i|j} \right) \quad (23)$$

The mixture of multinomials model for text assumes that a document is generated by first choosing a component j, and then drawing a number of words with

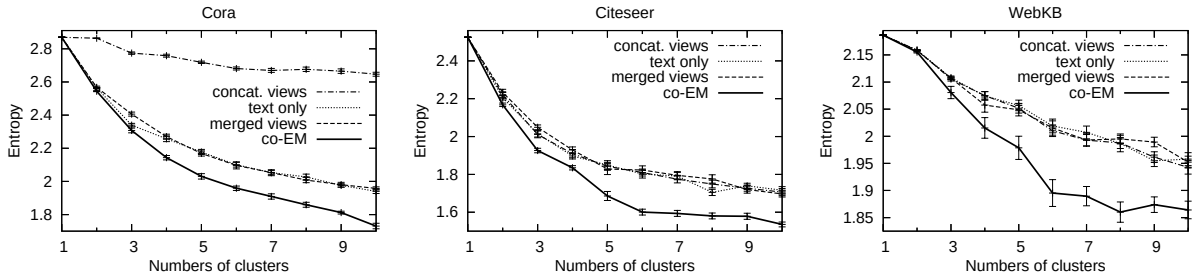


Figure 1. Average cluster impurity over varying numbers of clusters.

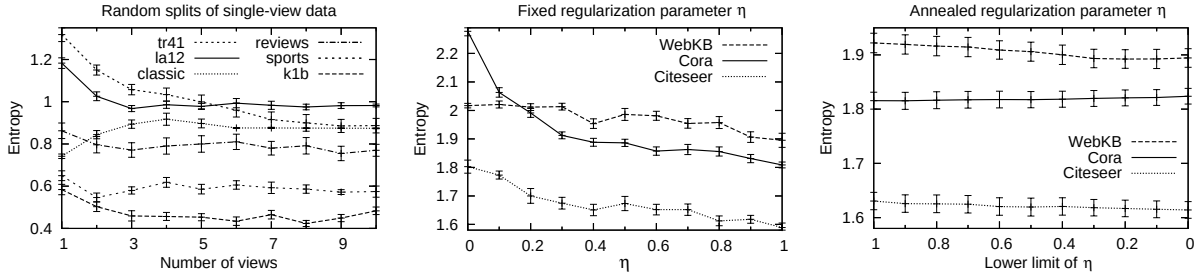


Figure 2. Six single-view data sets with random feature splits into views (left); tuning the regularization parameter η to a fixed value (center); annealing η during the optimization process (right).

replacement according to a component-specific likelihood. The multinomial link model analogously assumes that, for a document x, a number of references *from* or *to* other documents are drawn according to a component-specific likelihood. We first use three sets of linked documents for our experimentation. The *Citeseer* data set contains 3,312 entries that belong to six classes. The text view consists of title and abstract of a paper; the two link views are inbound and outbound references. The *Cora* data set contains 9,947 computer science papers categorized into eight classes. In addition to the three views of the Citeseer data set we extract an anchor text view that contains three sentences centered at the occurrence of the reference in the text. The *WebKB* data set is a collection of 4,502 academic web pages manually grouped into six classes. Two views contain the text on the page and the anchor text of all inbound links, respectively. The total number of views are 2 (WebKB), 3 (Citeseer), and 4 (Cora).

Note that web pages or publications do not necessarily have inbound or outbound links. We require only the title/abstract and web page text views to contain attributes. The other views are empty in many cases; the inbound link view of 45% of the Cora instances is empty. In order to account for this application-specific property, we include only non-empty views in the averaged posterior $P^{(v)}(j|x_i, \Theta_t, \eta)$.

We use two single-view baselines. The first baseline applies single-view EM to a concatenation of all views (caption "concat. views"). The second baseline merges all text views (anchor text and intrinsic text are merged into one bag) and separately merges all link views (corresponding to an undirected graphical model). Single-view EM is than applied to the concatenation of these views ("merged views"). All results are averaged over 20 runs and error bars indicate standard error. Figure 1 details the clustering performance of the algorithm and baselines for various numbers of clusters (mixture components assumed by the model). Co-EM outperforms the baselines for all problems and any number of clusters.

In order to find out how multi-view co-EM performs when there is no natural feature split in the data, we randomly draw six single-view document data sets that come with the cluto clustering toolkit (Zhao & Karypis, 2001). We randomly split the available attributes into s subsets and average the performance over 20 distinct attribute splits. We set the number of clusters to the respective number of true mixture components. Figure 2 (left) shows the results for several numbers of views. We can see that in all but one case the best number of views is greater than one. In four of six cases we can reject the null hypothesis that one view incurs a lower entropy than two views at a significance level of $\alpha = 0.01$. Additionally, in 2 out of six cases, three views lead to significantly better clusters than two views; in four out of six cases, the entropy has its empirical minimum for more than two views.

In all experiments so far, we fixed $\eta = 1$. Let us study whether tuning or annealing η improves the cluster quality. Figure 2 (center) shows the entropy for various

fixed values of η; we see that 1 is the best setting ($\eta > 1$ would imply negative word probabilities $\theta_{lj}^{(v)}$).

Let us finally study whether a fixed value of η or annealing η results in a better cluster quality. In the following experiments, η is initialized at 1 and slowly annealed towards 0. Figure 2 (right) shows the development of the cluster entropy as η approaches towards 0. We see that fixing and annealing η empirically works equally well; annealing η causes a slight improvement in two cases and a slight deterioration of the quality in one case. The distinction between co-EM with and without annealing of η lies in the fact that convergence can only be proven when η is annealed; empirically, these variants are almost indistinguishable.

6. Conclusion

The Q^{MV} function defined in Equation 7 augments the single-view optimization criterion Q^{SV} by penalizing disagreement among distinct views. This is motivated by the result that the consensus among independent hypotheses upper-bounds the error rate of either hypothesis. Theorem 1 rewrites the criterion $Q^{MV}(\Theta, \Theta_t)$ such that it can easily be maximized over Θ when Θ_t is fixed: an M step is executed locally in each view. Maximizing Q^{MV} naturally leads to a version of the co-EM algorithm for arbitrarily many views and unlabeled data. Our derivation thus explains, motivates, and generalizes the co-EM algorithm.

While the original co-EM algorithm cannot be shown to converge, a variant of the method that anneals η over time can be guaranteed to converge and to (locally) maximize $\sum_v P(X^{(v)}|\Theta)$. Initially amplifying the convex error bound Δ in the criterion Q^{MV} serves as a heuristic that guides the search towards a better local optimum.

Our experiments show that co-EM is a better clustering procedure than single-view EM for actual multi-view problems such as clustering linked documents. Surprisingly, we also found that in most cases the impurity of text clusters can be reduced by splitting the attributes at random and applying multi-view clustering. This indicates that the consensus maximization principle may contribute to methods for a broader range of machine learning problems.

Acknowledgment

This work has been supported by the German Science Foundation DFG under grant SCHE540/10-1.

References

Abney, S. (2002). Bootstrapping. *Proc. of the 40th Annual Meeting of the Association for Comp. Linguistics.*

Barnard, K., Duygulu, P., Forsyth, D., de Freitas, N., Blei, D., & Jordan, M. (2002). Matching words and pictures. *Journal of Machine Learning Research, 3*, 1107–1135.

Becker, S., & Hinton, G. (1992). A self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature, 355*, 161–163.

Bickel, S., & Scheffer, T. (2004). Multi-view clustering. *Proc. of the IEEE International Conf. on Data Mining.*

Blei, D., & Jordan, M. (2003). Modeling annotated data. *Proceedings of the ACM SIGIR Conference on Information Retrieval.*

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the Conference on Computational Learning Theory.*

Brefeld, U., & Scheffer, T. (2004). Co-EM support vector learning. *Proc. of the Int. Conf. on Machine Learning.*

Collins, M., & Singer, Y. (1999). Unsupervised models for named entity classification. *Proc. of the Conf. on Empirical Methods in Natural Language Processing.*

Dasgupta, S., Littman, M., & McAllester, D. (2001). PAC generalization bounds for co-training. *Proceedings of Neural Information Processing Systems.*

de Sa, V. (1994). Learning classification with unlabeled data. *Proc. of Neural Information Processing Systems.*

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B, 39.*

Ghani, R. (2002). Combining labeled and unlabeled data for multiclass text categorization. *Proceedings of the International Conference on Machine Learning.*

Kailing, K., Kriegel, H., Pryakhin, A., & Schubert, M. (2004). Clustering multi-represented objects with noise. *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining.*

McCallum, A., & Nigam, K. (1998). Employing EM in pool-based active learning for text classification. *Proc. of the International Conference on Machine Learning.*

Muslea, I., Kloblock, C., & Minton, S. (2002). Active + semi-supervised learning = robust multi-view learning. *Proc. of the International Conf. on Machine Learning.*

Nigam, K., & Ghani, R. (2000). Analyzing the effectiveness and applicability of co-training. *Proceedings of the Workshop on Information and Knowledge Management.*

Sinkkonen, J., Nikkilä, J., Lahti, L., & Kaski, S. (2004). Associative clustering. *Proceedings of the European Conference on Machine Learning.*

$$\sum_{v\neq u}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)},\Theta_t^{(v)})\log\frac{P(x_i^{(v)}|\Theta^{(v)})}{P(x_i^{(u)}|\Theta^{(u)})} = \sum_{v\neq u}\sum_{i=1}^{n}\log\frac{P(x_i^{(v)}|\Theta^{(v)})}{P(x_i^{(u)}|\Theta^{(u)})} \tag{24}$$

$$= \sum_{v<u}\sum_{i=1}^{n}\left(\log\frac{P(x_i^{(v)}|\Theta^{(v)})}{P(x_i^{(u)}|\Theta^{(u)})}+\log\frac{P(x_i^{(u)}|\Theta^{(u)})}{P(x_i^{(v)}|\Theta^{(v)})}\right) = \sum_{v<u}\sum_{i=1}^{n}\log\frac{P(x_i^{(v)}|\Theta^{(v)})P(x_i^{(u)}|\Theta^{(u)})}{P(x_i^{(u)}|\Theta^{(u)})P(x_i^{(v)}|\Theta^{(v)})} = \sum_{v<u}\sum_{i=1}^{n}\log 1 = 0 \tag{25}$$

$$\Delta(\Theta^{(1)},\ldots,\Theta^{(s)},\Theta_t^{(1)},\ldots,\Theta_t^{(s)})$$

$$= \frac{1}{s-1}\sum_{v\neq u}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)},\Theta_t^{(v)})\log\frac{P(j|x_i^{(v)},\Theta^{(v)})}{P(j|x_i^{(u)},\Theta^{(u)})} + \frac{1}{s-1}\sum_{v\neq u}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)},\Theta_t^{(v)})\log\frac{P(x_i^{(v)}|\Theta^{(v)})}{P(x_i^{(u)}|\Theta^{(u)})} \tag{26}$$

$$= \frac{1}{s-1}\sum_{v\neq u}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)},\Theta_t^{(v)})\log\frac{P(x_i^{(v)},j|\Theta^{(v)})}{P(x_i^{(u)},j|\Theta^{(u)})} = \frac{1}{s-1}\sum_{v\neq u}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)},\Theta_t^{(v)})\log\frac{\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)})}{\alpha_j^{(u)}P(x_i^{(u)}|j,\Theta^{(u)})} \tag{27}$$

$$= \sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)},\Theta_t^{(v)})\log(\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)})) - \frac{1}{s-1}\sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m}\left(\sum_{\bar{v}} P(j|x_i^{(\bar{v})},\Theta_t^{(\bar{v})})\right)\log(\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)})) \tag{28}$$

$$Q^{MV}(\Theta^{(1)},\ldots,\Theta^{(s)},\Theta_t^{(1)},\ldots,\Theta_t^{(s)})$$

$$= \sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)},\Theta_t^{(v)})\log(\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)})) - \eta\sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m} P(j|x_i^{(v)},\Theta_t^{(v)})\log(\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)}))$$
$$+\frac{\eta}{s-1}\sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m}\left(\sum_{\bar{v}} P(j|x_i^{(\bar{v})},\Theta_t^{(\bar{v})})\right)\log(\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)})) \tag{29}$$

$$= \sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m}\log(\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)}))\left((1-\eta)P(j|x_i^{(v)},\Theta_t^{(v)}) + \frac{\eta}{s-1}\sum_{\bar{v}} P(j|x_i^{(\bar{v})},\Theta_t^{(\bar{v})})\right) \tag{30}$$

$$= \sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m} P^{(v)}(j|x_i,\Theta_t,\eta)\log(\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)})) \tag{31}$$

$$= \sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m} P^{(v)}(j|x_i,\Theta_t,\eta)\log\alpha_j^{(v)} + \sum_{v=1}^{s}\sum_{i=1}^{n}\sum_{j=1}^{m} P^{(v)}(j|x_i,\Theta_t,\eta)\log P(x_i^{(v)}|j,\Theta^{(v)}) \tag{32}$$

Wang, J., Zeng, H., Chen, Z., Lu, H., Tao, L., & Ma, W. (2003). Recom: Reinforcement clustering of multi-type interrelated data objects. *Proceedings of the ACM SIGIR Conference on Information Retrieval*.

Wu, J. (1983). On the convergence properties of the EM algorithm. *The Annals of Statistics*, *11*, 95–103.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. *Proc. of the Annual Meeting of the Association for Comp. Ling.*.

Zhao, Y., & Karypis, G. (2001). *Criterion functions for document clustering: Experiments and analysis* (Technical Report TR 01-40). Department of Computer Science, University of Minnesota, Minneapolis, MN.

Appendix

A. Proof of Theorem 1

In order to prove Theorem 1 we first prove two additional equations. Firstly, we prove that the left term of Equation 24 equals zero. Equation 24 holds because $\sum_{j=1}^{m} P(j|x_i,\Theta_t)\log C = \log C$ when C is independent of j. Instead of summing over all two-way combinations of views we sum only once over each pairwise combination in the left term of Equation 25 and merge the logarithms. The terms in the resulting fraction cancel to one.

Secondly, we simplify the dissent function Δ. In Equation 26 we add a term (Equation 24) that we proved to be zero in Equation 25. Equations 27 merge the logarithms and apply the chain rule to extract $\alpha_j^{(v)}$. In Equation 28 the logarithm is split up and the sum over all pairwise view combinations is substituted with a nested sum.

Now we can prove Theorem 1. We write Q^{MV} as a sum of single-view Q^{SV} criteria (Equation 5) and the transformed Δ of Equation 28, resulting in Equation 29. With Equation 30 the $\log(\alpha_j^{(v)}P(x_i^{(v)}|j,\Theta^{(v)}))$ terms are factored out. We introduce the abbreviation $P^{(v)}(j|x_i,\Theta_t,\eta)$ in Equation 31. Finally the logarithm is split up (Equation 32) and the proof is finished. □

Spectral Clustering with Two Views

Virginia R. de Sa DESA@UCSD.EDU
Department of Cognitive Science, 0515
University of California, San Diego
9500 Gilman Dr.
La Jolla, CA 92093-0515

Abstract

In this paper we develop an algorithm for spectral clustering in the multi-view setting where there are two independent subsets of dimensions, each of which could be used for clustering (or classification). The canonical examples of this are simultaneous input from two sensory modalitites, where input from each sensory modality is considered a view, as well as web pages where the text on the page is considered one view and text on links to the page another view. Our spectral clustering algorithm creates a bipartite graph and is based on the "minimizing-disagreement" idea. We show a simple artifically generated problem to illustrate when we expect it to perform well and then apply it to a web page clustering problem. We show that it performs better than clustering in the joint space and clustering in the individual spaces when some patterns have both views and others have just one view.

Spectral clustering is a very successful idea for clustering patterns. The idea is to form a pairwise affinity matrix A between all pairs of patterns, normalize it, and compute eigenvectors of this normalized affinity matrix (graph Laplacian)L. It can be shown that the second eigenvector of the normalized graph Laplacian is a relaxation of a binary vector solution that minimizes the normalized cut on a graph (Shi & Malik, 1998; J.Shi & Malik, ; Meila & Shi, 2001; Ng et al., 2001). Spectral clustering has the advantage of performing well with non-Gaussian clusters as well as being easily implementable. It is also non-iterative with no local minima. The Ng,Jordan,Weiss(Ng et al., 2001) (NJW) generalization to multiclass clustering (which we will build on) is summarized below for data patterns x_i to be clustered in to k clusters.

- Form the affinity matrix $A(i,j) = exp(-||x_i - x_j||^2/2\sigma^2)$
- Set the diagonal entries $A(i,i) = 0$
- Compute the normalized graph Laplacian as $L = D^{-.5}AD^{-.5}$ where D is a diagonal matrix with $D(i,i) = \sum_j A(i,j)$
- Compute top k eigenvectors of L and place as colums in a matrix X
- Form Y from X by normalizing the rows of X
- Run kmeans to cluster the row vectors of Y
- pattern x_i is assigned to cluster α iff row i of Y is assigned to cluster α

In this paper we develop an algorithm for spectral clustering in the multi-view setting where there are two independent subsets of dimensions, each of which could be used for clustering (or classification). The canonical examples of this are multi-sensory input from two modalities where input from each sensory modality is considered a view as well as web pages where the text on the page is considered one view and text on links to the page another view. Also computer vision applications with multiple conditionally independent sensor or feature vectors can be viewed in this way.

1. Algorithm Development

Our spectral multi-view algorithm is based on ideas originally developed for the (non-spectral) Minimizing-Disagreement algorithm (de Sa, 1994a; de Sa & Ballard, 1998). The idea behind the Minimizing-Disagreement (M-D) algorithm is that two (or more)

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

networks receiving data from different views, but with no explicit supervisory label, should cluster the data in each view so as to minimize the disagreement between the clusterings. The Minimizing-Disagreement algorithm was described intuitively using the following diagram shown in Figure 1. In the figure imagine that there are two classes of objects, with densities given by the thick curve and the thin curve and that this marginal density is the same in each one-dimensional view. The scatter plots on the left of the figure show two possible scenarios for how the "views" may be related. In the top case, the views are conditionally independent. Given that a "thick/dark" object is present, the particular pattern in each view is independent. On the right, the same data is represented in a different format. In this case the values in view 1 are represented along one line and the values in view 2 along another line. Lines are joined between a pair if those values occurred together. The minimizing disagreement algorithm wants to find a cut from top to bottom that crosses the fewest lines – within the pattern space (subject to some kind of balance constraint to prevent trivial solutions with empty or near empty clusters). Disagreement is minimized for the dashed line shown. Here we transform this intuitive idea for 1-D views to a general algorithm on a weighted bipartite graph.

The difficulty in transforming this intuitive idea into a general algorithm for a M-D spectral algorithm is that in describing it as making a cut from top to bottom, we assume that we have a neighborhood relationship within each top set and bottom set, that is not explicitly represented. That is we assume that points drawn in a line next to each other are similar points in the same view. Treating the points as nodes in a graph and applying a graph cut algorithm, would lose that information.

One solution would be to simply connect co-occurring values **and** also join nearest neighbors (or join neighbors according to a similarity measure) in each view. This, however, raises the tricky issue of how to encode the relative strengths of the pairing weights with the within-view affinity weights.

Instead, our solution is to draw reduced weight co-occurrence relationships between neighbors of an observed pair of patterns (weighted by a unimodal function such as a Gaussian). We call our algorithm **sM-D** Each input in each view is represented by a node in the graph. The strength of the weight between two nodes in different views depends on the number of multi-view patterns (which we can think of as co-occuring pairs of patterns) that are sufficiently close (in both views) (with a fall off in weight as the distances grow). This representation has the semantics that we believe there is noise in the actual patterns that occur or alternatively that we wish to consider the pairings simultaneously at multiple scales.

More specifically, let us define $x_i^{(v)}$ as view v of the ith pattern. We will construct a graph node for each view of each pattern and define $n_{(i,v)}$ to represent the node for view v of the ith pattern. Now consider the pattern $x_1^{(1)} = [1\ \ 2\ \ 1]'$ (where throughout this paper ' denotes the transpose operator) and the pattern $x_2^{(1)} = [1\ \ 2\ \ 1]' + \vec{\epsilon}'$. These two patterns should probably be considered identical for small $\vec{\epsilon}$. This means that $x_1^{(2)}$ the co-occurring pattern for $x_1^{(1)}$ should probably also be linked with $x_2^{(1)}$. The Gaussian weighting allows us to do this in a smooth way for increasing $\vec{\epsilon}$. To compute the total weight between node $n_{(i,1)}$ and $n_{(j,2)}$ we sum over all observed pattern co-occurrences (k=1 to p): the product of (the (Gaussian weighted) distance between $x_i^{(1)}$ (the pattern represented by $n_{(i,1)}$) and $x_k^{(1)}$ and the same same term for the relationship between the $x_j^{(2)}$ and $x_k^{(2)}$. That is

$$w_{ij} = \sum_p e^{-\frac{||(x_i^{(1)} - x_k^{(1)})||^2}{2\sigma_1^2}} e^{-\frac{||(x_j^{(2)} - x_k^{(2)})||^2}{2\sigma_2^2}} \quad (1)$$

$$= [A_{v1} \times A_{v2}]_{ij} \quad (2)$$

where A_{v1} is the affinity matrix for the view 1 patterns and A_{v2} the affinity matrix for just the view 2 patterns. $A_{v1}(i,j) = e^{\frac{-||(x_i^{(1)} - x_j^{(1)})||^2}{2\sigma_1^2}}$. Note that the product between the Gaussian weighted distances within each view is just the Gaussian weighted normalized distance between the two concatenated patterns (when considered as multi-view patterns).

Then we take the $p \times p$ matrix of w's and put it in a large $2p \times 2p$ matrix of the form

$$A_{sM-D} = \begin{bmatrix} 0_{p\times p} & W \\ W' & 0_{p\times p} \end{bmatrix}$$

where $0_{p\times p}$ represents a $p \times p$ matrix of zeros (and we will drop the subscript from here on for clarity). This matrix could then be considered an affinity matrix (for a bipartite graph) and given to the spectral clustering algorithm of (Ng et al., 2001). However note that the next step is to compute eigenvectors of the matrix

$$D^{-.5} A_{sM-D} D^{-.5}$$

where D is a diagonal matrix with $D(i,i) = \sum_j A_{sM-D}(i,j)$ (row sums of A_{sM-D}) which is equal

to (where D_{row} (D_{col}) is the diagonal matrix with diagonal entries equal to the row (column) sums of W)

$$\begin{bmatrix} D_{row}^{-.5} & 0 \\ 0 & D_{col}^{-.5} \end{bmatrix} \begin{bmatrix} 0 & W \\ W' & 0 \end{bmatrix} \begin{bmatrix} D_{row}^{-.5} & 0 \\ 0 & D_{col}^{-.5} \end{bmatrix}$$

but that matrix has the same eigenvectors as the matrix

$$\begin{bmatrix} D_{row}^{-.5} W D_{col}^{-1} W' D_{row}^{-.5} & 0 \\ 0 & D_{col}^{-.5} W' D_{row}^{-1} W D_{col}^{-.5} \end{bmatrix}$$

which has conjoined eigenvectors of each of the blocks $D_{row}^{-.5} W D_{col}^{-1} W' D_{row}^{-.5}$ and $D_{col}^{-.5} W' D_{row}^{-1} W D_{col}^{-.5}$ and these parts can be found efficiently together by computing the SVD of the matrix $L_W = D_{row}^{-.5} W D_{col}^{-.5}$. This trick is used in the co-clustering literature (Dhillon, 2001; Zha et al., 2001), but there the affinity submatrix W is derived simply from the term document matrix (or equivalent) not derived as a product of affinity matrices from different views [1]. The final clustering/segmentation is obtained from the top eigenvectors. There are several slightly different ways to cluster the values of this eigenvector. We use the prescription of Ng, Jordan and Weiss from the first page where Y is obtained as follows.

```
Av1 =exp(-distmatview1/(2*sigsq1));
Av2 =exp(-distmatview2/(2*sigsq2));
W=Av1*Av2;
Dtop=(sum(W'));
Dbot=(sum(W));
Lw=diag(Dtop.^(-.5))*W*diag(Dbot.^(-.5));
[U,S,V]=svds(Lw)
X=[U(:,1:numclusts);V(:,1:numclusts)];
Xsq=X.*X;
divmat=repmat(sqrt(sum(Xsq')'),1,numclusts);
Y=X./divmat;
```

Note that computing the SVD of the matrix $L_W = D_{row}^{-.5} W D_{col}^{-.5}$, gives two sets of eigenvectors, those of $L_W L_W'$ and those of $L_W' L_W$. The algorithm above concatenates these to form the matrix Y (as one would get if performing spectral clustering on the large matrix A_{sM-D}). This thus provides clusters for each view of each pattern. To get a cluster for the multi-view pattern, when both views are approximately equally reliable, the top p rows of the Y matrix can be averaged with the bottom p rows before the k-means step. If one view is significantly more reliable than the other, one can just use the Y entries corresponding to the more reliable view (The eigenvectors of $L_W L_W'$ reveal the clustering for the view 1 segments and the eigenvectors of $L_W' L_W$ for the view 2 segments). .

[1] It is possible to combine these ideas and use multiple views, each (or one) of which is a co-clustering

For comparison, we consider the patterns to be in the joint space given by the inputs in the two views. We call this algorithm **JOINT** In this case, we can simply use the standard spectral clustering algorithm to determine clusters. Note that in this case

$$\begin{aligned} A_{JOINT}(i,j) &= e^{-\frac{||(x_i - x_j)||^2}{2\sigma^2}} \\ &= e^{-\frac{||(x_i^{(1)} - x_j^{(1)})||^2 + ||x_i^{(2)} - x_j^{(2)})||^2}{2\sigma^2}} \\ &= A_{v1}(i,j)^{\frac{\sigma_1^2}{\sigma^2}} \cdot A_{v2}(i,j)^{\frac{\sigma_2^2}{\sigma^2}} \end{aligned}$$

Thus the affinity matrix for clustering in the joint space can be obtained by a componentwise product (Hadamard product or .* in Matlab) of the affinity matrices for the individual modalities. [As shown above, a person who ignored the multi-view structure of the data would use one σ^2 for all dimensions, however to give this algorithm the best chance we allowed the use of different σ_1^2 and σ_2^2.] In other words, we actually used $A_{JOINT}(i,j) = A_{v1}(i,j) \cdot A_{v2}(i,j)$

We also compare our algorithm to one where the affinity matrices of the two individual modalities are added. This idea is mentioned in (Joachims, 2003) for the semi-supervised case. We call this algorithm **SUM**. case $A_{SUM}(i,j) = A_{v1}(i,j) + A_{v2}(i,j)$.

2. Theoretical Comparison of Algorithms

As discussed in (Ng et al., 2001), the simplest case for spectral clustering algorithms, is when the affinity matrix is block diagonal. One can easily see that the following statements are true.

Statement 1: For consistent block diagonal A_{v1} and A_{v2}, all 3 algorithms preserve block diagonal form.

Statement 2: If the affinity matrix in one view is block diagonal but random in the other then only **JOINT** results in a block diagonal affinity matrix.

When is the **sM-D** algorithm better than the **JOINT** algorithm? Figure 2 shows a simple example that shows that clustering in the joint space and M-D style algorithms are not identical. The datapoints are numbered for the purposes of discussion. Consider in particular the membership of the circled datapoint (4). The **sM-D** algorithm would cluster it with datapoints 1, 2 and 3. The **JOINT** algorithm is much more likely (over a wider range of parameters and noise levels) to cluster datapoint 4 with datapoints 5,6,7 and 8. To quantify this effect, we constructed an affinity matrix for each view from the example in Figure 2 and

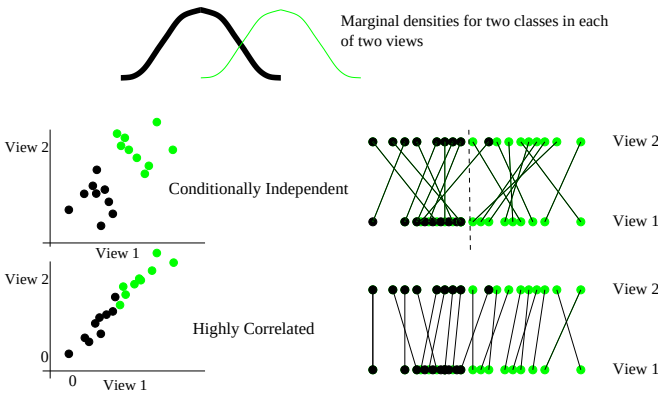


Figure 1. **A** Consider two classes of patterns with two 1-D views. The top of the figure represents the density for the two pattern classes (bold and unbold) in View 1. Assume the marginal densities in View 2 are similar. An example scatterplot is shown on the left of the figure. On the right,the same data is presented in a different format. Here lines are joined between co-occurring patterns in the two imaginary 1-D views/modalities (as shown at top). The M-D algorithm wants to find a partition that crosses the fewest lines. Two cases are shown for when the views are conditionally independent or highly correlated. In the conditionally independent case, there is a clear non-trivial optimal cut. In the correlated case,there are many equally good cuts and the M-D algorithm will not perform well in this case.

ran spectral clustering algorithms on noisy versions of these affinity matrices for varying levels of noise and varying cross-cluster strength m.

$$A_{v1} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & m & 0 & m & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & m & 0 & m & 0 \\ 0 & m & 0 & m & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & m & 0 & m & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

$$A_{v2} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & m & m & 0 & 0 \\ 0 & 0 & 1 & 1 & m & m & 0 & 0 \\ 0 & 0 & m & m & 1 & 1 & 0 & 0 \\ 0 & 0 & m & m & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

The cross-cluster strength m relates to the relative spacing between the two clusters with respect to the σ^2 parameter in the spectral clustering algorithm. The results are robust over a broad range of noise levels (10^{-19} to 10^{-1}). For m=0, all three algorithms correctly cluster nodes 1-4 and 5-8. However for $m \geq .05$ the **JOINT** method breaks down and groups one of nodes 4 or 5 with the wrong cluster. The **SUM** algorithm breaks down for $m \geq .81$ and the **sM-D** algorithm continues to group appropriately until $m = .92$. Figure 3 explains these results graphically as well as showing the actual (pre-noise) matrices computed W_{sM-D},A_{SUM}, and A_{JOINT}.

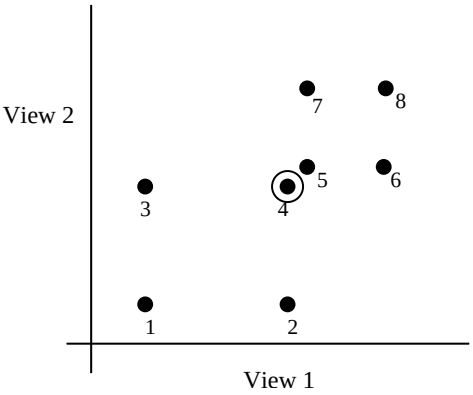


Figure 2. A simple example that would give a different solutions clustered in the joint space **JOINT**, than if the **sM-D** algorithm was used.

3. Clustering results with the course webpage dataset

This dataset consists of two views of web pages. The first view consists of text on the web page and the second view consists of text on the links to the web page (Blum & Mitchell, 1998). We use the six class (course, department, faculty, project, staff, student) version in (Bickel & Scheffer, 2004) consisting of tfidf (term frequency inverse document frequency - where a document is stored as a vector of weighted words. Tfidf weights words more if they occur more in a document and downweights words that occur often in the full dataset) vectors without stemming. Patterns were normalized within each view so that squared distances reflected the commonly used cosine similarity measure.

We use the average entropy error metric of (Bickel & Scheffer, 2004)

$$E = \sum_{i=1}^{k} \frac{m_i(-\sum_j p_{ij} \log_2(p_{ij}))}{m}$$

where p_{ij} is the proportion of cluster i that is from mixture component j, m_i is the number of patterns in class i and m is the total number of patterns. On this dataset, with this error measure, perfect agreement would result in $E = 0$, everybody in the same

class would give $E = 2.219$ (and equal size clusters with probability measurement equal to the base class probabilities also gives $E = 2.2$).

We first compared the algorithms on the full dataset. To do this we first searched for good σ_1 and σ_2 from clustering in the individual views.

We found that (with the proper normalization), the joint method worked slightly better(E=1.64) than the sum (E=1.70) and m-d version (E=1.66) (standard error estimates are provided later when 90% of the data is used). For comparison, Bickel and Scheffer report measures on the same error measure (with 6 clusters) of approximately[2] 1.73 (multi-view) and 2.03 (single view) for their mixture-of-multinomials EM algorithm and approximately 1.97 (multi-view) and 2.07 (single view) for their spherical k-Means algorithm (Bickel & Scheffer, 2004).

As mentioned, when computing the SVD of the matrix $L_W = D_{row}^{-.5} W D_{col}^{-.5}$, one gets two sets of eigenvectors, those of $L_W L_W'$ and those of $L_W' L_W$ and for equally reliable views, the Y matrices can be averaged before the k-means step. For this dataset however, view 1 is significantly more reliable than view 2 and we obtain improved performance by simply using the eigenvectors from view 1.

The main advantage of our algorithm is that it can allow us to combine sources of information with different numbers of views. To see this, remember that the affinity submatrix W is in terms of how similar pairs are to co-occurring pairs. Thus a single view pattern $x_i^{(1)}$ from view 1 does not contribute to the library of paired occurrences but can still be related to patterns $x_j^{(2)}$ in view 2 according to how similar the pair $(x_i^{(1)}, x_j^{(2)})$ is to the set of co-occurring patterns. Thus we can construct a full bipartite affinity matrix between patterns from view 1 and those from view 2 using equation 2 where p sums over only the paired patterns. This results in a matrix multiplication of the form $A_{v1} \times A_{v2}$ where this time A_{v1} is $(p+m) \times p$ dimensional and A_{v2} is $p \times (p+n)$ dimensional where there are p co-occurring (multi-view) patterns and m patterns with only view 1 and n patterns with only view 2 (see Figure 4). Note that the bottom right quadrant of the resulting W matrix computes the affinity between an unpaired view 1 pattern and an unpaired view 2 pattern according to the sum of the affinities between this pair $(x_{p+i}^{(1)}, x_{p+j}^{(2)})$ and each of the set of observed pairs $\{(x_1^{(1)}, x_1^{(2)}), ...(x_p^{(1)}, x_p^{(2)})\}$. The affinity between two pairs of patterns is the product between the affinity between each view of each pattern.

In this case we use the eigenvectors of $L_W L_W'$ to find the clusters for both the paired and view 1 data and must use the eigenvectors of $L_W' L_W$ to find the clusters for the data that only has view 2.

For comparison, we consider two other alternatives for clustering data that consists of some multi-view patterns and some single view patterns.

Alternative A using JOINT: cluster only the p patterns consisting of $x_i^{(1)}$ and $x_j^{(2)}$ concatenated in the joint space. Spectral clustering will give clusters for these patterns. To report clusters for the $m+n$ unpaired patterns, report the cluster of the nearest same view paired pattern of the pattern.

Alternative B: cluster the patterns from each view separately. In this case the pairing information is lost.

Results for different values of p are reported in Tables 1 thru 3. Table 1 shows that there is a very slight but significant performance advantage for the multi-view patterns using Alternative A when 2084 (90%) of the patterns have both views, but that Alternatives B and our sM-D method perform significantly better on the patterns that only have values for view 1 and our sM-D method performs significantly better than both alternatives for patterns that only have values for view 2. When only 1158 (50%) of the patterns are provided with two views, the sM-D algorithm performs significantly better in all categories. Table 3 shows how the sM-D algorithm varies for different numbers of paired patterns. (The slight improvement in clustering performance (with increased variance) for the paired view data in the 50% paired case is likely due to an increased chance of not including inappropriate pairs in the paired dataset. Performance decreases with non independent sources of information have been observed with the non-spectral M-D algorithm. If leaving out some data vectors increases the independence between views, we would expect improved performance.) Performance for the single view data is seen to decrease gradually with less paired training data.

One value of an algorithm that can train with multi-view data and report data for single-view data would be when the single-view data arrive at a later time. We are working on using the Nystrom approximation (Charless Fowlkes & Malik, 2004) for such out of sample estimates. This would allow us to train with paired data and provide cluster labels for later unpaired data.

[2] estimated from their graph

Table 3. Average Entropy for **sM-D** for varying amount of two-view data. (See Table 1 for an explanation of terms)

	2084 (90%)	**1621 (70%)**	**1158 (50%)**	**694 (30%)**	**231 (10%)**
both views	$1.68 \pm .003$	$1.66 \pm .006$	$1.64 \pm .01$	$1.68 \pm .01$	$1.76 \pm .03$
View 1 only	$1.63 \pm .02$	$1.66 \pm .01$	$1.66 \pm .006$	$1.67 \pm .01$	$1.73 \pm .02$
View 2 only	$1.83 \pm .02$	$1.91 \pm .01$	$1.95 \pm .006$	$1.97 \pm .01$	$2.00 \pm .01$

Table 1. Average Entropy where 2084 (90%) of the Patterns have both views. Alt. is an abbreviation for Alternative. All values are given $\pm$ 1 standard error of the mean over 10 runs. The both view line refers to the error for patterns that had two views, View 1 only refers to errors on patterns that consisted of only view 1 and View 2 only refers to errors on patterns that consisted of View 2 only. All errors are using the average entropy error measure

	Alt. A	**Alt B**	**sM-D**
both views	$1.66 \pm .003$	$1.68 \pm .002$	$1.68 \pm .003$
View 1 only	$1.83 \pm .02$	$1.64 \pm .02$	$1.63 \pm .02$
View 2 only	$1.95 \pm .02$	$2.04 \pm .003$	$1.83 \pm .02$

Table 2. Average Entropy where 1158 (50%) of the Patterns have both views. (See Table 1 for an explanation of terms)

	Alt. A	**Alt. B**	**sM-D**
both views	$1.67 \pm .01$	$1.69 \pm .002$	$1.64 \pm .01$
View 1 only	$1.90 \pm .02$	$1.68 \pm .006$	$1.66 \pm .006$
View 2 only	$2.04 \pm .006$	$2.04 \pm .003$	$1.95 \pm .006$

4. Discussion

We have shown that spectral clustering is competitive in the webpage domain and have introduced a novel multi-view spectral clustering algorithm. While it performs slightly worse than properly normalized joint spectral clustering in the full webpage domain, the difference is small and the sM-D algorithm has the major advantage that it allows single view patterns to benefit from the paired dataset. This allows one to incorporate all available information to form the best clusters when there is lots of single-view data to be clustered.

The spectral Minimizing-Disagreement algorithm was motivated by the earlier Minimizing-Disagreement algorithm(de Sa, 1994a; de Sa & Ballard, 1998) and we believe that of the different ways of spectral clustering with multiple views, sM-D best incorporates the idea of minimizing the disagreement of the outputs of two classifiers (clusterers). In the appendix we reproduce an argument from (de Sa, 1994b; de Sa & Ballard, 1998) that motivates, in the 1-D case, the minimizing-disagreement approach as an approximation to minimizing misclassifications.

The spectral implementation of the Minimizing-Disagreement idea shares many of the advantages and disadvantages of other spectral techniques. It does not work as well for multi-class classifications as for binary. It is quick to implement and run (with sparse matrices) and has a guaranteed global optimum which is related by a relaxation to the desired optimum.

Putting the algorithm in the framework of graph partitioning should allow easier comparison and combination with results from clustering in the joint space. Also it should be straightforward to modify the algorithm to incorporate some labeled data so that the algorithm can be used in a semi-supervised way. We are currently exploring these avenues.

Appendix: Minimizing Disagreement as an Approximation to Minimizing Misclassifications

The M-D algorithm to minimize the disagreement corresponds to the LVQ2.1 algorithm(Kohonen, 1990) except that the "label" for each view's pattern is the hypothesized output of the other view. To understand how making use of this label, through minimizing the disagreement between the two outputs, relates to the true goal of minimizing misclassifications in each view, consider the conditionally independent (within a class) version of the 2-view example illustrated in Figure 5. In the supervised case (Figure 5A) the availability of the actual labels allows sampling of the actual marginal distributions. For each view, the number of misclassifications can be minimized by setting the boundaries for each view at the crossing points of their marginal distributions.

However in the self-supervised system, the labels are not available. Instead we are given the output of the

other view. Consider the system from the point of view of view 2. Its patterns are labeled according to the outputs of view 1. This labels the patterns in Class A as shown in Figure 5B. Thus from the actual Class A patterns, the second view sees the "labeled" distributions shown. Letting a be the fraction of Class A patterns that are misclassified by view 1, the resulting distributions of the real Class A patterns seen by view 2 are $(1-a)P(C_A)p(x_2|C_A)$ and $(a)P(C_A)p(x_2|C_A)$.

Similarly Figure 5C shows View 2's view of the patterns from class B (given View 1's current border). Letting b be the fraction of Class B patterns misclassified by view 1, the distributions are given by $(1-b)P(C_B)p(x_2|C_B)$ and $(b)P(C_B)p(x_2|C_B)$. Combining the effects on both classes results in the "labeled" distributions shown in Figure 5D. The "apparent Class A" distribution is given by $(1-a)P(C_A)p(x_2|C_A) + (b)P(C_B)p(x_2|C_B)$ and the "apparent Class B" distribution by $(a)P(C_A)p(x_2|C_A)+(1-b)P(C_B)p(x_2|C_B)$. The crossing point of these two distributions occurs at the value of x_2 for which $(1-2a)P(C_A)p(x_2|C_A) = (1-2b)P(C_B)p(x_2|C_B)$. Comparing this with the crossing point of the actual distributions that occurs at x_2 satisfying $P(C_A)p(x_2|C_A) = P(C_B)p(x_2|C_B)$ reveals that if the proportion of Class A patterns misclassified by view 1 is the same as the proportion of Class B patterns misclassified by view 1 (i.e. $a = b$) the crossing points of the distributions will be identical. This is true even though the approximated distributions will be discrepant for all cases where there are any misclassified patterns ($a > 0$ OR $b > 0$). If $a \approx b$, the crossing point will be close.

Simultaneously the second view is labeling the patterns to the first view. At each iteration of the algorithm both borders move according to the samples from the "apparent" marginal distributions.

Acknowledgments

Many thanks to Ulf Brefeld, Tobias Scheffer, Steffen Bickel, and Anjum Gupta for kindly sending their processed datasets and to Marina Meila and Deepak Verma for providing a great library of spectral clustering code at http://www.cs.washington.edu/homes/deepak/spectral/library.tgz. Finally, warm thanks to Patrick Gallagher, Jochen Triesch, Serge Belongie and three anonymous reviewers for helpful comments. This work is supported by NSF CAREER grant 0133996.

References

Bickel, S., & Scheffer, T. (2004). Multi-view clustering. *Proceedings of the IEEE International Conference on Data Mining.*

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory (COLT-98)* (pp. 92–100). Madison, WI.

Charless Fowlkes, Serge Belongie, F. C., & Malik, J. (2004). Spectral grouping using the nystrom method. *IEEE Transactions Pattern Analysis and Machine Intelligence, 26.*

de Sa, V. R. (1994a). Learning classification with unlabeled data. *Advances in Neural Information Processing Systems 6* (pp. 112—119). Morgan Kaufmann.

de Sa, V. R. (1994b). *Unsupervised classification learning from cross-modal environmental structure.* Doctoral dissertation, Department of Computer Science, University of Rochester. also available as TR 536 (November 1994).

de Sa, V. R., & Ballard, D. H. (1998). Category learning through multimodality sensing. *Neural Computation, 10,* 1097–1117.

Dhillon, I. S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. *KDD 2001.* San Francisco, CA.

Joachims, T. (2003). Transductive learning via spectral graph partitioning. *Proceedings of the 20th International Conference on Machine Learning (ICML 2003).*

J.Shi, & Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 888–905.

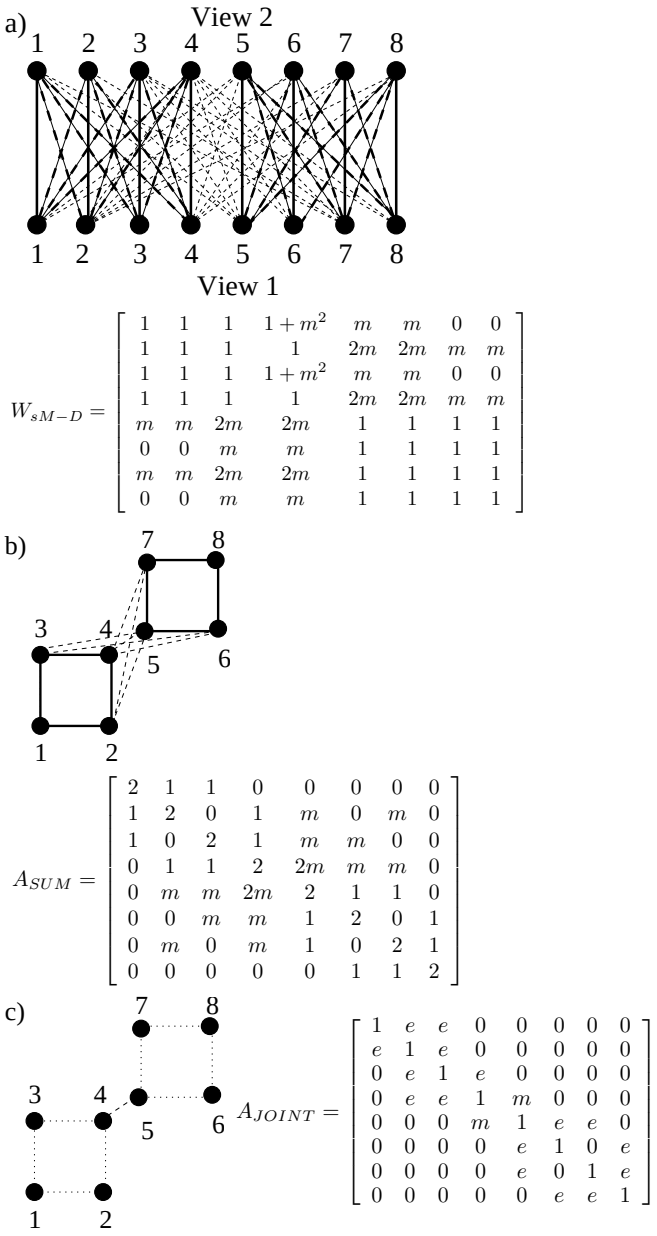
Kohonen, T. (1990). Improved versions of learning vector quantization. *IJCNN International Joint Conference on Neural Networks* (pp. I–545–I–550).

Meila, M., & Shi, J. (2001). Learning segmentation by random walks. *Advances in Neural Information Processing Systems 13.*

Ng, A. Y., Jordan, M. I., & Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems 14.*

Shi, J., & Malik, J. (1998). Motion segmentation and tracking using normalized cuts. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Zha, H., Ding, C., & Gu, M. (2001). Bipartite graph partitioning and data clustering. *CIKM '01.*



$$W_{sM-D} = \begin{bmatrix} 1 & 1 & 1 & 1+m^2 & m & m & 0 & 0 \\ 1 & 1 & 1 & 1 & 2m & 2m & m & m \\ 1 & 1 & 1 & 1+m^2 & m & m & 0 & 0 \\ 1 & 1 & 1 & 1 & 2m & 2m & m & m \\ m & m & 2m & 2m & 1 & 1 & 1 & 1 \\ 0 & 0 & m & m & 1 & 1 & 1 & 1 \\ m & m & 2m & 2m & 1 & 1 & 1 & 1 \\ 0 & 0 & m & m & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$A_{SUM} = \begin{bmatrix} 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & m & 0 & m & 0 \\ 1 & 0 & 2 & 1 & m & m & 0 & 0 \\ 0 & 1 & 1 & 2 & 2m & m & m & 0 \\ 0 & m & m & 2m & 2 & 1 & 1 & 0 \\ 0 & 0 & m & m & 1 & 2 & 0 & 1 \\ 0 & m & 0 & m & 1 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 2 \end{bmatrix}$$

$$A_{JOINT} = \begin{bmatrix} 1 & e & e & 0 & 0 & 0 & 0 & 0 \\ e & 1 & e & 0 & 0 & 0 & 0 & 0 \\ 0 & e & 1 & e & 0 & 0 & 0 & 0 \\ 0 & e & e & 1 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & m & 1 & e & e & 0 \\ 0 & 0 & 0 & 0 & e & 1 & 0 & e \\ 0 & 0 & 0 & 0 & e & 0 & 1 & e \\ 0 & 0 & 0 & 0 & 0 & e & e & 1 \end{bmatrix}$$

Figure 3. The resulting graphs (and matrices) resulting from the three algorithms a) sM-D b)SUM c) JOINT. applied to the matrices A_1 and A_2 above. The light lines correspond to weights of m and 2m and the dark lines correspond to weights of 1 and $1 + m^2$. In a) the solid lines correspond to co-occurrence lines and the dashed lines, inferred relationships. In c) the faint dotted lines only arise due to noise. The e's in the matrix result only from the noise and would be different small numbers at each spot). **Each algorithm tries to find the smallest normalized cut in its graph**

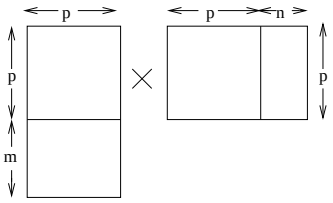


Figure 4. A graphical view of the matrix multiplication required to compute W when there are p patterns with both views, m patterns with only view 1 and n patterns with only view 2.

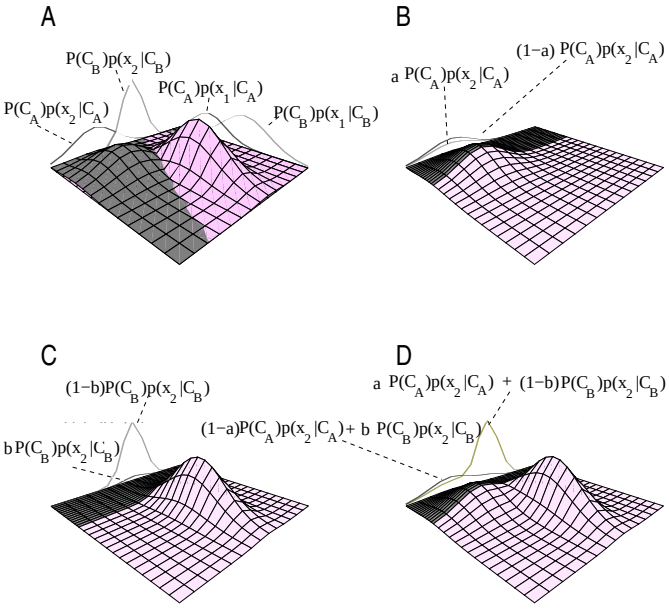


Figure 5. **An example joint and marginal distribution for a conditionally independent example problem**. (For better visualization the joint distribution is expanded vertically twice as much as the marginal distributions.) The darker gray represents patterns labeled "A", while the lighter gray are labeled "B". (A) shows the labeling for the supervised case. (B) shows the labeling of Class A patterns as seen by view 2 given the view 1 border shown. ***a*** represents the fraction of the Class A patterns that are misclassified by view 1. (C) shows the labeling of Class B patterns as seen by view 2 given the same view 1 border. ***b*** represents the fraction of the Class B patterns that are misclassified by view 1. (D) shows the total pattern distributions seen by view 2 given the labels determined by view 1. These distributions can be considered as the labeled distributions on which view 2 is performing a form of supervised learning. (However it is more complicated as view 1's border is concurrently influenced by the current position of view 2's border). See text for more details.

The use of machine translation tools for cross-lingual text mining

Blaž Fortuna BLAZ.FORTUNA@IJS.SI
Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia

John Shawe-Taylor JST@ECS.SOTON.AC.UK
University of Southampton, Southampton SO17 1BJ

Abstract

Eigen-analysis such as LSI or KCCA was already successfully applied to cross-lingual information retrieval. This approach has a weakness in that it needs an aligned training set of documents. In this paper we address this weakness and show that it can be successfully avoided through the use of machine translation. We show that the performance is similar on the domains where human generated training seta are available. However for other domains artificial training sets can be generated that significantly outperform human generated ones obtained from a different domain.

1. Introduction

The use of eigen-analysis in cross-lingual information retrieval was pioneered by Dumais et al. (Dumais et al., 1996). They used Latent Semantic Indexing to documents formed by concatenating the two versions of each document into a single file. The training set was therefore required to be a paired dataset, meaning a set of documents together with their translations into the second language.

This restriction also applied to the later application of kernel canonical correlation analysis to this task (Vinokourov et al., 2002). The difference in this approach is that the two versions of the documents are kept separate and projection directions for the two languages are sought that maximise the correlation between the projections of the training data. These directions are then used to create a 'semantic space' in which the cross-lingual analysis is performed.

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

This approach was applied initially to the Hansard corpus of English/French paired documents from the Canadian parliament (Vinokourov et al., 2002). The semantic space derived in this way was further used to perform text classification on a separate corpus. Here the Reuters-21578 data was used.

The same approach has been used for more distinct languages in a paper studying cross-lingual information retrieval of Japanese patents (Li & Shawe-Taylor, 2005). Again this relied on using a paired dataset of Japanese patents as training data.

The approach to cross-lingual information retrieval and semantic representation has therefore proven reliable and effective in a number of different contexts. There is, however, an apparently unavoidable weakness to the approach in that a paired training set is required whose documents adequately cover the topics of interest. Indeed in the experiment that applied the semantic space learned with Hansard data to the Reuter's documents, the small overlap of the two vocabularies inevitably resulted in poorer performance.

This paper addresses this weakness by using machine translation to generate paired datasets that can be used to derive a semantic space using documents directly relevant to the target domain.

The paper is organised as follows. The next section discusses the questions raised by the use of automatic translation and outlines the experiments that will be presented to provide answers to these questions. Section 3 gives a brief summary of the KCCA approach to finding a semantic subspace mapping, while Section 4 presents the experimental results. We finish with some conclusions.

2. Using machine translation

The use of machine translation (MT) ensures that appropriate datasets can be generated but raises the

question of whether their quality will be sufficient to derive an accurate semantic space. Clearly we would expect that having a hand translated dataset will be preferable to using MT software. The first question this paper will address is the extent to which this is true.

Hence, the paper investigates how the quality of a machine translation generated dataset compares with a true paired corpus when one is available. This experiment is performed on the Hansard corpus with very encouraging results.

The advantage of using a machine generated paired dataset is that the topic of the articles will be identical to those on which the analysis is to be performed. In contrast the best available hand translated corpus might be for documents whose topics are only loosely related to those being studied. So we have a dilemma: do we use a machine translated corpus with a close topic match or a hand translated corpus with a weaker match. The second set of experiments reported in this paper will attempt to address this dilemma.

We consider a dataset for which paired training data is not available. Here we tackle a classification task and investigate the effectiveness of the semantic space generated from the translated paired corpus. We compare classification accuracy using this space with the space obtained from a paired dataset with a weaker overlap of topic with the documents being classified. For these experiments we used the now standard classification algorithm of support vector machines. Again the results obtained are very encouraging.

2.1. Related work

The MT was already used in the context of cross-language IR. D. W. Oard used it in the (D. W. Oard, 1998) as a method for translating the queries or the documents between bag-of-words spaces for different languages. A more similar approach to ours was used in the (M. L. Littman, S. T. Dumais and T. K. Landauer, 1998). They generated a separate LSI semantic space for each of the languages. For example, the semantic space was generated using the English documents from the training set and all non-English documents from the test set were than translated using MT and mapped into this semantic space. Our approach differs in that it only uses MT for the training period. In a practical setup this can be crucial since there is no need to call the time-expencive MT in the query loop. The aim of this paper is to show that MT can be used for obtaining a paired corpus for KCCA that is well matched to the target documents and not to to perform a general comparison of KCCA with other CLIR methods.

3. Summary of KCCA

Canonical Correlation Analysis (CCA) is a method of correlating two multidimensional variables. It makes use of two different views of the same semantic object (eg. the same text document written in two different languages) to extract representation of the underlying semantics.

Input to CCA is a paired dataset $S = \{(u_i, v_i); u_i \in U, v_i \in V\}$, where U and V are two different views of the data – each pair contains two views of the same document. The goal of CCA is to find two linear mappings into a common semantic space W from the spaces U and V. All documents from U and V can be mapped into W to obtain a view- or in our case language-independent representation.

The criterion used to choose the mapping is the correlation between the projections of the two views across the training data in each dimension. This criterion leads to a generalised eigenvalue problem whose eigenvectors give the desired mappings.

CCA can be kernelized so it can be applied to feature vectors only implicitly available through a kernel function. There is a danger that spurious correlations could be found in high dimensional spaces and so the method has to be regularised by constraining the norms of the projection weight vectors. A parameter τ controls the degree of regularisation introduced. The kernelized version is called Kernel Canonical Correlation Analysis (KCCA).

Example Let the space V be the vector-space model for English and U the vector-space model for French text documents. A paired dataset is then a set of pairs of English documents together with their French translation. The output of KCCA on this dataset is a semantic space where each dimension shares similar English and French meaning. By mapping English or French documents into this space, a language independent-representation is obtained. In this way standard machine learning algorithms can be used on multi-lingual datasets.

4. Experiments

In the following experiments, two issues regarding artificially generated corpora are discussed. First we compared it to a human generated corpus in domains where a human generated corpus is already available. The goal of this part is to check if the artificial corpus

can deliver comparable results. For the second part of the experiments we chose a domain and a problem for which human generated corpora were not available. We wanted to show, that by using documents from this domain an artificial corpus can be generated which outperforms human generated corpora obtained from different domains. Due to the datasets available we chose an information retrieval task for the first part of experiments and a text classification task for the second part.

4.1. Information Retrieval

The first part of experiments was done on the Hansards corpus (Germann, 2001). This corpus contains around 1.3 million pairs of aligned text chunks from the official records of the 36th Canadian Parliament. The raw text was split into sentences with Adwait Ratnaparkhi's MXTERMINATOR and aligned with I. Dan Melamed's GSA tool. The corpus is split into two parts, House Debates (around 83% of text chunks) and Senate Debates. These parts are than split into a training part and two testing parts. For our experiments we used the House Debates part from which we used only the training part and first testing part. The text chunks were split into 'paragraphs' based on '* * *' delimiters and these paragraphs were treated as separate documents. We only used documents that had the same number of lines in both their English and French version.

The training part was used as a human generated aligned corpus for learning semantic space with KCCA. In order to generate an artificial aligned corpus we first split the training documents into two halves. From the first half we kept only the English documents and only the French documents from the second half. In this way we obtained two independent sets of documents, one for each language. We then used *Google Language Tools* [1] to translate each document into its opposite language and generate an artificial aligned corpus. Some statistics on the corpora used in this experiment can be found in Table 1.

Table 1. Hansards aligned corpora

	Train	Artificial	Test1
Text chunks	495,022	495,022	42,011
Documents	9,918	9,918	896
En. words	38,252	39,395	16,136
Fr. words	52,391	55,425	21,001

From each corpus we learned with KCCA a language independent semantic space with 400, 800 or 1200 dimensions on a subset of 1500 documents.

[1] `http://www.google.com/language_tools`

Table 2. Top1 and Top10 results for the queries with 5 keywords are on left side and with 10 keywords are on the right side

n	1 [%]	10 [%]	1 [%]	10 [%]
En - En	96	100	99	100
Fr - Fr	97	100	100	100

The documents for these subsets were selected randomly and all results were averaged over five runs with different seeds for the random number generator. We ran experiments for the regularization parameter τ set to 0.2, 0.5 and 0.8, but because results for different parameters were not much different only results for $\tau = 0.5$ are presented. The threshold for the Partial Gram-Schmidt algorithm (or equivalently incomplete Cholesky decomposition of the kernel matrices) was set to 0.4.

For the information retrieval task, the entire first testing part of the Hansards corpus was projected into the language independent semantic space learned from the human generated corpus or from the artificial corpus. Each query was treated as a text document and its TFIDF vector was projected into the KCCA semantic space. Testing documents were than retrieved using nearest neighbour with cosine distance to the query.

In the first experiment each English document was used as a query and only its mate document in French was considered relevant for that query (Vinokourov et al., 2002). The same was done with French documents as queries and English documents as test documents. We measured the number of times that the relevant document appeared in the set of the top n retrieved documents (Top n). The Top1 results for both corpora are on average 96-98%, with results for human generated corpus generally scoring around 2% higher. The Top10 results were 100% for the both corpora.

For the next experiment we extracted 5 or 10 keywords from each document, according to their TFIDF weights, and used them for a query. Only the document from which the query was extracted and its mate document were regarded as relevant. We first tested queries in the original bag-of-words space and these results can serve as a baseline for the experiments done in the KCCA semantic spaces. Results are shown in Table 2. All queries were then tested in a similar way as before, the only difference is that this time we also measured the accuracy for cases where the language of the query and the relevant document were the same. Results for the queries with 5 keywords are presented

Table 3. Top1 and Top10 results for the queries with 5 keywords for the human generated corpus (top) and artificial corpus (bottom). The numbers are Top1/Top10 in percent.

	EN – EN	EN – FR	FR – EN	FR – FR
dim	1/10	1/10	1/10	1/10
400	76/98	59/93	60/92	74/98
800	83/99	64/95	65/94	81/99
1200	87/99	66/96	65/95	84/99
400	76/97	49/89	50/87	72/97
800	84/99	55/91	56/89	80/99
1200	86/99	58/91	59/90	83/99

in Table 3. and for the queries with 10 keywords in Table 4.

It is interesting to note that, for cases where the query was in the same language as the documents we searched over, the results are equal or slightly better for the artificial corpus than for the human generated one. This shows that, from both corpora, KCCA finds a similar semantic basis in vector-space models of English and French documents. However, the results for the artificial corpus are not as good as for the human generated corpus when it comes to cross-lingual queries. For queries with only 5 keywords, Top1 results for the artificial corpus are on average around 8% lower than for the human generated corpus while for queries with 10 keywords this drops to around 7%. Note that this difference stays constant when the dimensionality of semantic space increases. The difference between artificial and human generated corpora, when measuring the recall for the top 10 retrieved documents, drops to around 5% for queries with 5 keywords and to only 2% for queries with 10 keywords. The results for the cross-language parts of the experiments are lower for the artificial corpus than for the human generated corpus. The difference is not significant and a language independent semantic space learned on an artificial aligned corpus can still be successfully used in practice.

Table 4. Top1 and Top10 results for the queries with 10 keywords for the human generated corpus (top) and artificial corpus (bottom). The numbers are Top1/Top10 in percent.

	EN – EN	EN – FR	FR – EN	FR – FR
dim	1/10	1/10	1/10	1/10
400	93/99	79/99	78/97	90/100
800	96/100	82/99	81/98	94/100
1200	97/100	82/99	81/98	96/100
400	94/100	70/96	69/96	91/100
800	97/100	75/98	75/97	95/100
1200	97/100	77/98	75/97	96/100

4.2. Classification

The second part of the experiments was done on the Reuters multilingual corpora (Reuters, 2004) (mul, 2004), which contain articles in English, French, German, Russian, Japanese and other languages. Only articles in English, French and German were used for this experiment. Articles for each language were collected independently and no human generated aligned corpus was available for this domain. All articles are annotated with categories.

The task addressed in this experiment was how to make use of the existing corpus of annotated documents from one language, for example English, for doing classification in some other language, for example French. This can be done with the use of KCCA for construction of a language independent semantic space in which annotated English documents can be used to train a classifier that can also be applied to the French documents [4]. The problem with this approach is that the expensive task of annotating French documents is replaced with the even more expensive task of generating the aligned corpus needed for KCCA. This can be elegantly avoided through the use of MT tools. Another possibility is to use an aligned corpus from some other domain, for example the Hansards corpus used in the previous experiments. However, documents from that corpus belong to different domain and may not cover all the semantics that appear in the news articles. On the other hand the artificial corpus is constructed from the same set of documents that will be used for training the classifiers.

For this experiment we picked 5000 documents for each of the three languages from the Reuters corpus. Subsets of these documents formed the training datasets for the classifiers. These same documents were also used for generating artificial aligned corpora in the same way as in the first part of the experiments; Google Language Tools were used to translate English documents to French and German and the other way around. In this way we generated English-French and English-German aligned corpora. We used the training part of the Hansards corpus as English-French human generated aligned corpora. Some statistics on the corpora used in this experiment can be found in Table 5.

KCCA was used for learning a language independent semantic space from these aligned corpora. The pa-

Table 5. English-French and English-German aligned corpora from the Reuters corpus.

	En-Fr	En-Gr
Paragraphs	119,181	104,639
Documents	10,000	10,000
English words	57,035	53,004
French words	66,925	—
German words	—	121,193

rameters used for learning were the same as for the information retrieval task. The only difference is that only subsets of 1000 documents were used. These subsets were selected randomly and the results presented were averaged over 5 runs. A linear SVM was used as a classification algorithm with cost parameter C set to 1.

The classification experiment was run in the following way. All the classifiers were trained on subsets of 3000 documents from the training set and the results were averaged over 5 runs. This means that the presented results are averaged over 25 runs. The classifiers trained in the original vector-space models are used as a baseline to which the ones trained in the KCCA semantic space can be compared. The documents from the English training set were projected into KCCA semantic space and a classifier was trained on them. The same was done with the French and German documents. The classifiers were tested on a subset of 50,000 documents from the Reuters corpora. The testing documents were also projected to KCCA semantic space for classifiers living in that space. We measured average precision: baseline results are shown in Table 6.

Table 6. Average precision for classifiers for categories CCAT, MCAT, ECAT and GCAT

	CCAT	MCAT	ECAT	GCAT
English	85 %	80 %	62 %	86 %
French	83 %	85 %	63 %	94 %
German	85 %	86 %	62 %	91 %

The results for the human generated and for the artificial English-French corpus are presented in Table 7 and in Figure 1. The results obtained with the artificial corpus were in all cases significantly better than the results obtained with the Hansard corpus and are close to the baseline results for single language classification. Note that the results for the artificial corpus only slightly improve when the dimensionality of semantic space increases from 400 to 800 while the results for the human generated corpus increase by 10 or more percent. This shows that the first 400 dimensions learned from the artificial corpus are much richer at capturing the semantics of news articles than the ones learned from Hansard corpus.

Table 7. Average precision [%] for classifiers learned in KCCA semantic space learned Hansards/artificial corpus (Hansard/artificial). The results are for the semantic space wiht 400 (top) and 800 (bottom) dimensions.

	CCAT	MCAT	ECAT	GCAT
En-En	59/79	40/76	25/51	51/78
En-Fr	41/78	21/81	18/54	75/89
Fr-En	55/80	30/76	22/50	40/77
Fr-Fr	40/78	24/82	19/54	77/89
En-En	67/80	61/82	38/54	67/79
En-Fr	47/79	32/82	27/55	80/90
Fr-En	60/80	43/76	30/52	51/78
Fr-Fr	53/79	59/83	38/56	85/89

Results for classification based on English-German artificial aligned corpus are shown in Table 8. Surprisingly in some cases the cross-lingual classifications do better than a straight German classifier. The results are not as close to the base line (Table 6) as the results from English-French artificial corpus. We suspect that this is due to the different structure of German which is evident in Table 5; the number of different words in the German articles is twice as high as in the English or French documents. One workaround would be to use more advanced preprocessing before using the bag

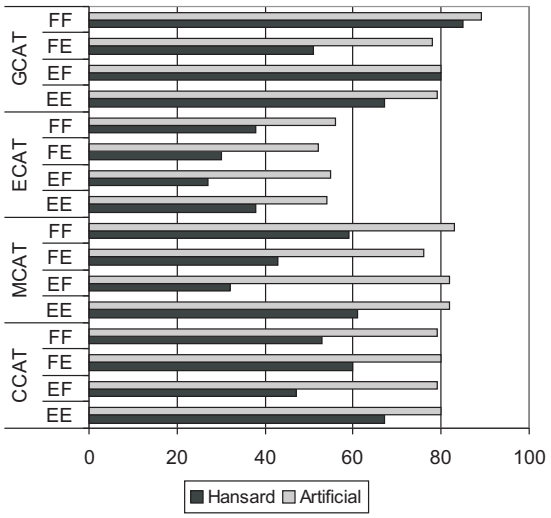


Figure 1. Average precision [%] for the classification of English and French documents in the 800 dimensional semantic space.

Table 8. Average precision [%] for classifiers learned in KCCA semantic space learned on artificially generated English-German aligned corpus

	CCAT	MCAT	ECAT	GCAT
En-En	75	77	49	81
En-Gr	72	82	46	87
Gr-En	70	75	43	78
Gr-Gr	67	83	44	86
En-En	76	78	52	82
En-Gr	73	82	47	88
Gr-En	71	75	46	79
Gr-Gr	68	83	47	86

of words or a use of different document representation like the string kernel.

5. Conclusion

The paper has addressed a pressing practical problem in the application of KCCA to cross-lingual information retrieval and language-independent semantic space induction in general, namely how to find an appropriate paired dataset.

Frequently we will only have access to a hand translated training corpus that is loosely related to the document corpus that is being analysed. The paper proposes a method of addressing this problem by using automatic machine translation tools to generate an 'artificial' paired corpus directly from the document corpus itself.

This raises two questions that are analysed in the paper. Firstly, how much worse is a semantic space derived from an artificial corpus than from a hand translated one, and secondly can the topic match offset any degradation resulting from the machine translation.

The first experiment showed that the degradation in performance does exist when we move to MT, but in a testing cross-lingual information retrieval task the reduction in recall was below 10%. This result certainly suggests that the advantage of exact topic match could well result in an increase in the quality of the semantic space obtained for a corpus with no hand translations available.

Our second experiment confirms this conjecture by demonstrating that the MT method improves cross-lingual classification results for the multi-lingual Reuters corpus when compared with using the semantic space induced from the hand translated Hansard corpus.

For these experiments the results are even more encouraging. They show a very significant advantage for the MT approach. Furthermore, the difference between the classification results using the semantic space and those obtained for single language classification using the bag of words feature space is not very large. This suggests that the method could be used to provide a general language independent classifier that can be used to classify documents from either language. This could potentially make it possible to use the topic labelling from one language to generate labels for newswire documents from the second language without the need for trained staff with appropriate language skills to perform the classification.

References

Reuters (2004). RCV2 - The Reuters Multi-lingual corpus.

Dumais, S. T., Landauer, T. K., & Littman, M. L. (1996). Automatic cross-linguistic information retrieval using latent semantic indexing. *Working Notes of the Workshop on Cross-Linguistic Information Retrieval*.

Germann, U. (2001). Aligned Hansards of the 36th Parliament of Canada. http://www.isi.edu/natural-language/download/hansard/. Release 2001-1a.

D. W. Oard (1998). A comparative study of query and document translation for cross-language information retrieval. *In proceedings of the 3rd Conference of the Association for Machine Translation in the Americas, pages 472-483*.

M. L. Littman, S. T. Dumais and T. K. Landauer (1998). Automatic Cross-Language Information Retrieval using Latent Semantic Indexing. *In cross-Language Information Retrieval, Kluwer.*.

Li, Y., & Shawe-Taylor, J. (2005). Using kcca for japanese-english cross-language information retrieval and classification. *to appear in Journal of Intelligent Information Systems*.

Reuters (2004). RCV1-v2/LYRL2004: The LYRL2004 Distribution of the RCV1-v2 Text Categorization Test Collection. http://jmlr.csail.mit.edu/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm.

Vinokourov, A., Shawe-Taylor, J., & Cristianini, N. (2002). Inferring a semantic representation of text via cross-language correlation analysis. *Advances of Neural Information Processing Systems 15*.

Invited Talk

Rayid Ghani RAYID.GHANI@ACCENTURE.COM
Accenture, 161 North Clark St, Chicago IL 60601, USA

(details of the talk to be determined)

Hybrid Hierarchical Clustering: Forming a Tree From Multiple Views

Anjum Gupta A3GUPTA@CS.UCSD.EDU
University of California, San Diego. 9500 Gilman Drive, La Jolla, CA 92093-0114

Sanjoy Dasgupta DASGUPTA@CS.UCSD.EDU
University of California, San Diego. 9500 Gilman Drive, La Jolla, CA 92093-0114

Abstract

We propose an algorithm for forming a hierarchical clustering when multiple views of the data are available. Different views of the data may have different underlying distance measures which suggest different clusterings. In such cases, combining the views to get a good clustering of the data becomes a challenging task. We allow these different underlying distance measures to be arbitrary Bregman divergences (which includes squared-Euclidean and KL distance). We start by extending the average-linkage method of agglomerative hierarchical clustering (Ward's method) to accommodate arbitrary Bregman distances. We then propose a method to combine multiple views, represented by different distance measures, into a single hierarchical clustering. For each binary split in this tree, we consider the various views (each of which suggests a clustering), and choose the one which gives the most significant reduction in cost. This method of interleaving the different views seems to work better than simply taking a linear combination of the distance measures, or concatenating the feature vectors of different views. We present some encouraging empirical results by generating such a hybrid tree for English phonemes.

1. Introduction

There has been a lot of recent machine learning research on exploiting multiple views of data. For instance, (Blum & Mitchell, 1998) notice that web pages can be viewed in two ways – by the words occurring in them, and by the words occurring in pages that point to them – and show how a certain type of conditional independence between these views can be exploited very effectively in semi-supervised learning. Likewise, (Collins & Singer, 1999) demonstrate a semi-supervised method for learning a named-entity classifier, using spelling and context as the two different views.

There has also been some encouraging work on using multiple views for unsupervised learning (eg. (Dhillon et al., 2003) and (Bickel & Scheffer, 2004)), in particular for clustering. It is natural to think that multiple views of the data should help yield better clusterings. However, there is a basic problem that needs to be resolved carefully. The various views may suggest rather different and incompatible clusterings of the data, especially if there is some independence between them. How can these different clusterings be reconciled?

We focus on hierarchical clusterings. These are popular tools for exploratory data analysis because they depict data at many level of granularity, and because there are simple algorithms for constructing them. In this paper, we propose a method for reconciling multiple views to generate a single hierarchical clustering.

Our model is as follows: there are n objects to be clustered. Each view corresponds to a different distance function on these objects. The most common distance function, which for instance underlies k-means, is squared Euclidean distance. Another useful measure is KL-divergence, which for instance is used for clustering words in (Pereira et al., 1993). These two distance functions are members of a much larger family, the Bregman divergences. These are the natural distance functions for exponential families of distributions, which is perhaps why they crop up in diverse machine learning contexts (Lafferty et al., 1997; Banerjee et al., 2004). In this paper, we allow each view to be

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

represented by a different Bregman divergence.

We start by extending average-linkage agglomerative clustering (specifically, Ward's method) to accommodate arbitrary Bregman divergences. Average-linkage typically assumes squared Euclidean norm as the underlying distance measure, and exploits special properties of this distance to substantially increase efficiency. We show that these same speedups can be realized for all Bregman divergences.

A straightforward way to accommodate multiple views would be to use a linear combination of their different distance measures. This approach runs into some basic problems. First of all, the different distances might at very different scales, and it might not be possible to make them comparable to one another by a simple linear transformation. Second, if the views represent very different information, then a linear combination of the two distances may simply serve to dampen or obscure the information in each. These intuitions are borne out in the experiments we conduct, in which linear combinations of the distance measures tend to destroy well-formed clusters that are present in individual views (based on just one distance function).

We propose a hybrid hierarchical clustering which is constructed top-down and in which each binary split is based upon a single view, the best view at that particular juncture. At each point in the tree construction, we have a certain cluster of objects that needs to be partitioned in two. We try out all the views; for each view, we determine a good split (into two clusters) using agglomerative clustering, and we note the reduction in cost due to that split. We choose the view that gives the biggest multiplicative decrease in cost. Thus, the tree keeps the best splits – the most significant clusterings – suggested by each view.

To try this out, we formed a hierarchical clustering for 39 phonemes, using data from the TIMIT database (Zue & Seneff, 1988). We used two views of each phoneme: a 39-dimensional vector in Euclidean space, the mean of the samples of that phoneme, where each speech sample is encoded using the standard mel-frequency cepstral coefficients. For the second view, we considered context information, specifically the distribution over the next phoneme. This is a probability vector in 39-dimensional space, with KL-divergence as the natural distance. The results were encouraging.

2. Bregman Divergences

Many of the most common families of probability distributions – such as Gaussian, Binomial, and Poisson – are *exponential families*. This formalism has turned out to be very powerful in statistics and machine learning because it is general enough to include many distributions of interest (another example: the distributions which factor over a specified undirected graph) while at the same time being specific enough that it implies all sorts of special properties.

It turns out that each exponential family has a natural distance measure associated with it. In the case of spherical Gaussians, it is perhaps obvious what this distance measure is: squared Euclidean distance, because the density at any given point is determined by its squared Euclidean distance from the mean.

Let's look at another example. In the multinomial distribution, it can be checked that the density of a point depends on its KL-divergence from the mean. In a crucial sense, therefore, KL divergence is the natural distance measure of the multinomial. Notice that it is not a *metric*: it is not symmetric and does not satisfy the triangle inequality. However, as we will see, it is well-behaved in some ways and has a lot in common with squared Euclidean distance.

The various distance measures underlying different exponential families are collectively known as the *Bregman divergences* (Lafferty et al., 1997; Banerjee et al., 2004). We now give the standard formal definition of these divergences, which does not follow the intuition about exponential families but rather associates each divergence with a specific convex function.

Definition Let $\phi : \mathcal{S} \to \mathbf{R}$ be a strictly convex function which is defined on a convex domain $\mathcal{S} \subset \mathbf{R}^d$ and is differentiable on the interior of $\mathcal{S}$. The Bregman distance $D_\phi : \mathcal{S} \times \text{int}(\mathcal{S}) \to [0, \infty)$ is then defined by

$$D_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - \nabla\phi(\mathbf{y}) \cdot (\mathbf{x} - \mathbf{y}). \quad (1)$$

Some examples: choosing $\phi = \frac{1}{2}\|\mathbf{x}\|^2$ gives $D_\phi(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2$, squared Euclidean distance.

$$\begin{aligned} D_\phi(\mathbf{x}, \mathbf{y}) &= \frac{1}{2}\|\mathbf{x}\|^2 - \frac{1}{2}\|\mathbf{y}\|^2 - \mathbf{y} \cdot (\mathbf{x} - \mathbf{y}). \\ &= \frac{1}{2}\|\mathbf{x}\|^2 - \frac{1}{2}\|\mathbf{y}\|^2 - \mathbf{y} \cdot \mathbf{x} + \|\mathbf{y}\|^2 \\ &= \frac{1}{2}\|\mathbf{x}\|^2 + \frac{1}{2}\|\mathbf{y}\|^2 - \mathbf{y} \cdot \mathbf{x} \\ &= \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2 \end{aligned} \quad (2)$$

Similarly, $\phi(\mathbf{x}) = \sum_{i=1}^d x_i \log x_i$ gives

$$D_\phi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^d x_i \log \frac{x_i}{y_i} - \sum_i x_i + \sum_i y_i, \quad (3)$$

which is a generalization of KL-divergence (it reduces to the regular definition when $\mathbf{x}$ and $\mathbf{y}$ are probability measures and therefore sum to one).

2.1. Properties of Bregman divergences

Bregman divergences share a lot of the special properties of squared Euclidean distance. For instance, they satisfy a Pythagorean theorem (Lafferty et al., 1997). This makes it hopeful that many algorithms which seem expressly designed for squared Euclidean distance (and therefore for data which is, in a sense, Gaussian), such as k-means or average-linkage clustering, might be extendable to other Bregman divergences (that is, to other exponential families).

Recent work (Banerjee et al., 2004) has extended k-means to arbitrary Bregman divergences. This is possible due to certain properties that *all* Bregman divergences possess:

1. Given any set of points S, the single point μ which minimizes the aggregated Bregman distance
$$\sum_{x \in S} D_\phi(x, \mu)$$
is simply the mean of S, which we'll denote μ_S.

2. The additional cost incurred by choosing a different point $\mu \neq \mu_S$ as the center of cluster S has a very simple form:
$$\sum_{x \in S} D_\phi(x, \mu) = \sum_{x \in S} D_\phi(x, \mu_S) + |S| \cdot D_\phi(\mu_S, \mu) \tag{4}$$

We will make extensive use of these properties.

3. Extending average-linkage clustering to Bregman divergences

There are several methods for average-linkage agglomerative clustering, of which Ward's method is perhaps the most principled. Given n data points, it starts by putting each point in a singleton cluster of its own. It then repeatedly merges the two closest clusters, until there is just one cluster containing all the points. The sequence of merges defines the hierarchical clustering tree; along the way we get k-clusterings (partitions into k clusters) for all $k = 1, \ldots, n$. In Ward's method, which is designed to use squared Euclidean distance, the distance between two clusters S, T is the increase in k-means cost occasioned by merging them, in other words, $\text{cost}(S \cup T) - \text{cost}(S) - \text{cost}(T)$, where the cost of a set of points is defined as

$$\text{cost}(S) \;=\; \sum_{x \in S} \|x - \mu_S\|^2.$$

In particular, when the algorithm has $k+1$ clusters, and is deciding which pair to merge, it will choose the pair whose merger gives the smallest overall k-means cost. Thus Ward's method strives to produce k-clusterings with small k-means cost, for all k.

This suggests how to extend the method to other Bregman divergences: simply change the cost function,

$$\text{cost}(S) \;=\; \sum_{x \in S} D_\phi(x, \mu_S).$$

Notice that this makes sense because of property 1 above. The resulting algorithm is once again trying to minimize k-means cost, but for our more general distance functions.

This is straightforward enough, but more work is needed to make this new algorithm practical. In Ward's method, the properties of Euclidean distance are used to compute the cost of candidate mergers,

$$Ward_{Euc}(S, T) = \text{cost}(S \cup T) - \text{cost}(S) - \text{cost}(T),$$

very quickly. There is no need to actually sum over these three clusters; instead, the expression reduces to

$$Ward_{Euc}(S, T) = \frac{|S||T|}{|S| + |T|} \|\mu_S - \mu_T\|^2,$$

which is very quick, assuming that the means and sizes of clusters are kept available. Notice also that $\mu_{S \cup T}$ and $|S \cup T|$ can easily be computed from the means and sizes of S, T.

We now see that a similar simplification is possible for any Bregman divergence, and thus we can handle any of these distance functions without any additional time complexity. More precisely:

Lemma For any Bregman divergence D_ϕ, the cost of merging two clusters S, T is:

$$\begin{aligned} Ward_\phi(S, T) \;&=\; \text{cost}(S \cup T) - \text{cost}(S) - \text{cost}(T) \\ &=\; |S|\phi(\mu_S) + |T|\phi(\mu_T) - (|S| + |T|)\phi(\mu_{S \cup T}). \end{aligned}$$

Proof: For any Bregman divergence D_ϕ, the cost of merging two clusters S, T is:

$$\begin{aligned} &Cost(S \cup T) - Cost(S) - Cost(T) \\ &\qquad = \sum_{x \in S \cup T} D_\phi(x, \mu_{S \cup T}) \;-\; \sum_{x \in S} D_\phi(x, \mu_s) \\ &\qquad\qquad -\; \sum_{x \in T} D_\phi(x, \mu_T) \end{aligned}$$

Using equation (4), we get

$$
\begin{aligned}
&= \sum_{x \in S \cup T} D_\phi(x, \mu_{S\cup T}) \\
&\quad - \sum_{x \in S} D_\phi(x, \mu_{S\cup T}) + |S| \cdot D_\phi(\mu_s, \mu_{S\cup T}) \\
&\quad - \sum_{x \in T} D_\phi(x, \mu_{S\cup T}) + |T| \cdot D_\phi(\mu_T, \mu_{S\cup T})
\end{aligned}
$$

$$
\begin{aligned}
&= |S| \cdot \phi(\mu_s) + |T| \cdot \phi(\mu_T) - (|S| + |T|) \cdot \phi(\mu_{S\cup T}) \\
&\quad - |S| \cdot \mu_s \phi'(\mu_{S\cup T}) + |S| \cdot \mu_{S\cup T} \phi'(\mu_{S\cup T}) \\
&\quad - |T| \cdot \mu_T \phi'(\mu_{S\cup T}) + |T| \cdot \mu_{S\cup T} \phi'(\mu_{S\cup T})
\end{aligned}
$$

$$
\begin{aligned}
&= |S|\phi(\mu_s) + |T|\phi(\mu_T) - (|S| + |T|)\phi(\mu_{S\cup T}) \\
&\quad - \left[\sum_{x \in S\cup T} x \right] \phi'(\mu_{S\cup T}) + \left[\sum_{x \in S\cup T} x \right] \phi'(\mu_{S\cup T})
\end{aligned}
$$

$$
= |S| \cdot \phi(\mu_s) + |T| \cdot \phi(\mu_T) - (|S| + |T|)\phi(\mu_{S\cup T})
$$

So we get,

$$
\begin{aligned}
&Cost(S \cup T) - Cost(S) - Cost(T) \\
&\quad = |S| \cdot \phi(\mu_s) + |T| \cdot \phi(\mu_T) - (|S| + |T|)\phi(\mu_{S\cup T})
\end{aligned}
$$

The final expression above can be evaluated quickly. For example, for KL distance, we get

$$
\begin{aligned}
Ward_{KL}(S,T) &= |S|\mu_S \cdot \log(\mu_S) + |T|\mu_T \cdot \log(\mu_T) \\
&- (|S| + |T|)\mu_{S\cup T} \cdot \log(\mu_{S\cup T})
\end{aligned}
$$

(where the logarithms are taken coordinatewise).

4. Multiple views

Often we may have different set of features, obtained in different ways, and giving different type of information about the data we are trying to cluster. For example, we can obtain three different views of a web page, the first being the words in the web page itself, the second being the words in the web pages pointing to it, and third being some other statistical information about the page such as the size, number of times it is accessed etc. Each of the views is a useful source of information about the web page, and together they should be able to yield a better clustering than we could get from one view alone, but it is not obvious how to combine them.

Hybrid clustering algorithm

Input: A set of n points given as two views (representations) X and Y, each in a space with an underlying Bregman divergence.

1. If $n = 1$ put the single point in a leaf and return.
2. Apply the generalized Ward's method to X and Y in turn, and in each case retrieve the 2-clustering. Call these C_{x1}, C_{x2} (for X) and C_{y1}, C_{y2} (for Y).
3. Choose the "better" of the two splits, to divide X, Y into two clusters, $[X_1, Y_1]$ and $[X_2, Y_2]$.
4. Create a tree node for this split.
5. Recursively handle $[X_1, Y_1]$ and $[X_2, Y_2]$.
6. Return tree.

4.1. Combining multiple views: first try

Perhaps the most obvious approach to accommodating multiple views is to concatenate the feature vectors corresponding to the different representations, or more generally, to use a linear combination of their different underlying distance measures. This approach is problematic on two fronts. First, the various distance measures might be incomparable (as with Euclidean distance and KL divergence), making it somewhat absurd to form linear combinations of them. Second, if the views are orthogonal and suggest different clusterings, then in the linear combination this structure might get obscured. In fact, in our experiments we see when features are naively combined by such methods, we lose some of the good clusters clearly present in visible in the individual views.

4.2. A hybrid approach

When combining multiple views, we want to preserve cluster structures that are strongly suggested by the individual views. The idea is that if there was a strong separation between the data points in one of views, that separation should not be lost while combining the information from other views. We propose building a hierarchical tree in a top-down fashion that uses the best view available at each split point in the tree. This hybrid algorithm is outlined in the figure above.

To choose the best view for a given split, the algorithm computes the 2-clusterings suggested by all the different views, and picks the best one. How should this be defined? Intuitively, we want to pick the view that provides the most well-separated clustering, that is, the largest reduction in the cost. We have to be

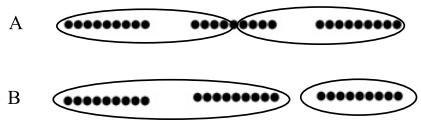


Figure 1. When there are three true clusters, K-means could split the third clusters into two pieces (shown in A), where as 2-clusters generated using Ward's method is more likely to keep the clusters intact (shown in B).

careful about how to do this; since the distance functions in different views are potentially on very different scales it is not a good idea to simply compare absolute differences in cluster costs.

Instead, we use ratios. We measure the goodness of a particular 2-clustering (binary split suggested by a particular view) by the ratio of its cost to the cost of the single combined cluster. That is, the goodness of a split of cluster X into $[X1, X2]$ is

$$\frac{\text{cost}(X)}{\text{cost}(X_1) + \text{cost}(X_2)}.$$

This particular measure (or rather its reciprocal) is equivalent to the (2-norm) Davies-Bouldin index (Davies & Bouldin, 1979) of similarity between two clusters.

When we are generating a 2-clustering from a particular view, we use agglomerative clustering (Ward's method) rather than k-means, even though the latter is simpler and quicker. Since at every step we split the data into exactly two clusters, using k-means could give us a bad division in case the data had three true clusters: it is more likely to split the third cluster into two pieces than Ward's method, which would likely instead split the three clusters by putting the two closer ones together(Figure 1). This approach adds a $O(n^2)$ complexity to our algorithm so may not be used for larger data sets.

5. Experiments

Our experimental results are for the 39 English language phonemes. We used two views of each phoneme, that were available in the TIMIT data set. The first view was intended to represent the speech signal itself, and consisted of a 39-dimensional vector in Euclidean space. To form this vector for a given phoneme, we looked at all utterances of that phoneme in the data set and transformed each utterance into a sequence of 39-dimensional vectors consisting of mel-frequency

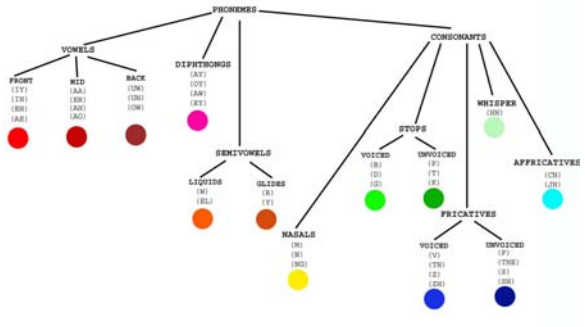


Figure 2. Classification chart of English phonemes, as in (Rabiner & Juang, 1993). Each family of phonemes has been assigned a color to make it easier to compare the various trees generated by experiments.

cepstral coefficients. This is a standard representation for speech recognizers. We picked one vector of coefficients from roughly the middle of each utterance, and then averaged these (over utterances) to get a single 39-dimensional vector for the phoneme. We thought of this vector as residing in Euclidean space since this is implicitly the distance measure used on this data by most speech recognizers.

The second view consisted of context information, specifically about the next phoneme. For each phoneme, we constructed a 39-dimensional vector representing transition probabilities to other phonemes (the i^{th} entry was the chance that this particular phoneme would be followed by the i^{th} phoneme). We used Laplace smoothing to avoid zero probability values. For this view, we used KL-divergence since it is a natural distance to use with probability measures. (It is purely a coincidence that both views of the phonemes have the same dimensionality.)

A reference hierarchical clustering already exists for phonemes, and in shown in Figure 2, copied over from (Rabiner & Juang, 1993). This provides an invaluable standard against which to judge the various hierarchical clusterings we generate.

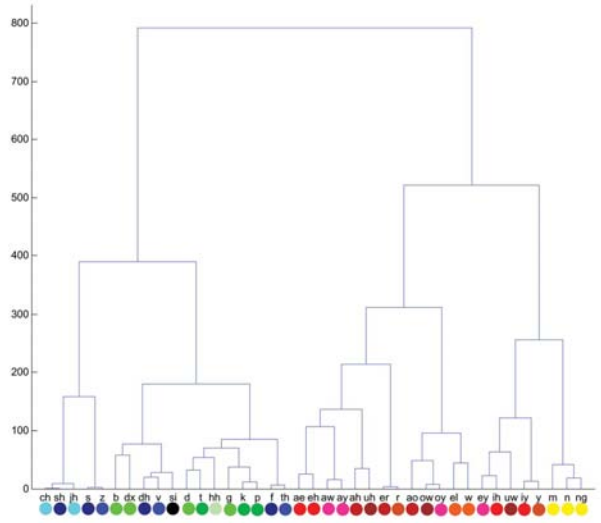


Figure 3. Hierarchical clustering based upon the speech signal, in 39-dimensional Euclidean space.

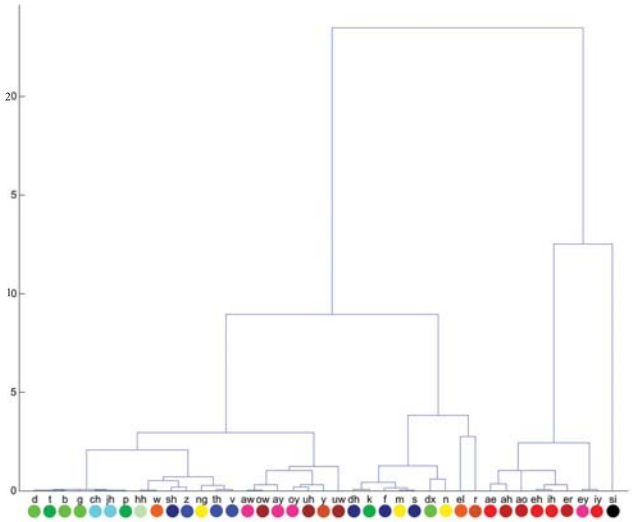


Figure 4. Hierarchical clustering based upon context information, with KL-divergence as the distance measure.

5.1. Pure hierarchical clusterings

We first generated hierarchical clusterings based on individual views. The first of these, using Euclidean distance on the speech signal representation, is shown in Figure 3. The second, using KL divergence on the context representation, is shown in Figure 4. In the trees produced by these individual views, there are a few clusters that partially match the clusters suggested by the reference classification in Figure 2. The first view does a good job of separating stops, fricatives and affricatives from the rest, although it is not good at distinguishing between these three groups. The second view is overall less competent, although it does a better job of distinguishing stops, fricatives, and affricatives. However, each view by itself is quite far from the reference clustering.

5.2. Way to combine multiple views

The trees generated by each individual view with their corresponding distance measures seem to complement each other by showing part of the whole picture. This motivated us to use both measures together to generate one consolidated tree. We first tried the obvious trick of combining the two distance measures by using a linear combination of the two distance measures. The result was still a Bregman divergence and thus amenable to our generalized agglomerative clustering scheme. The tree in Figure 5 is based upon a particular linear combination that tries to partially account for the different scales of the two distance measures.

Figure 5 doesn't improve on the clustering given by each individual view separately, and in fact demolishes some clusters suggested by KL distance.

5.3. Better way to combine the multiple views

Finally, we combined the two views into a hybrid tree, using the proposed algorithm. The result is in Figure 7. The hybrid tree manages to preserve the separations that were strongly suggested by each view, to unite the good points of each. A good separation of nasals, vowels, stops and fricatives from the first view and a better separation of stops and affricatives due to the second view is clearly present in the hybrid tree. The height of the tree nodes in the hybrid tree do not correspond to the closeness of the points under that node, since the algorithm works in a recursive way, the entire left tree is generated before the right tree.

Figure 6, shows the histogram of the similarity measure between the two clusters at every split in the hierarchy. Notice that *this is the inverse of the "goodness of split"*. The lower the similarity between clusters, the better the split. We observe that on the whole, the splits found by our hybrid algorithm lead to less similar pairs of clusters than those found by the regular Ward's algorithm using a single view. This corroborates our intuition that the significant separations in either view should make it into the hybrid view.

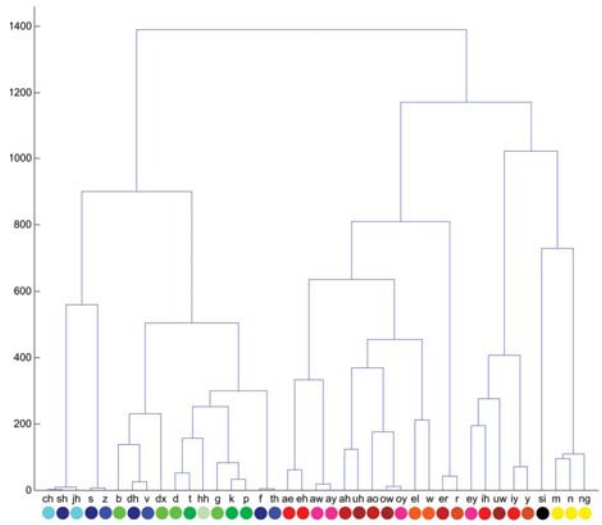


Figure 5. Hierarchical clustering based on a linear combination of the two distance functions. The tree looks very similar to the one built only in Euclidean space.

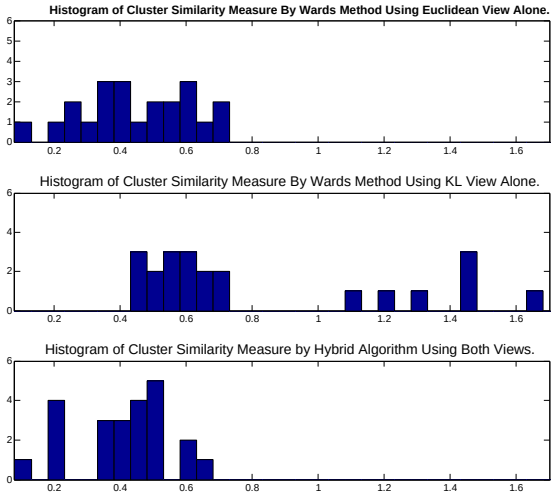


Figure 6. Histogram of similarity measures between the two clusters at every split in the hierarchy (smaller values are better). The distribution of these similarity values is noticeably better for the hybrid algorithm than for the other two.

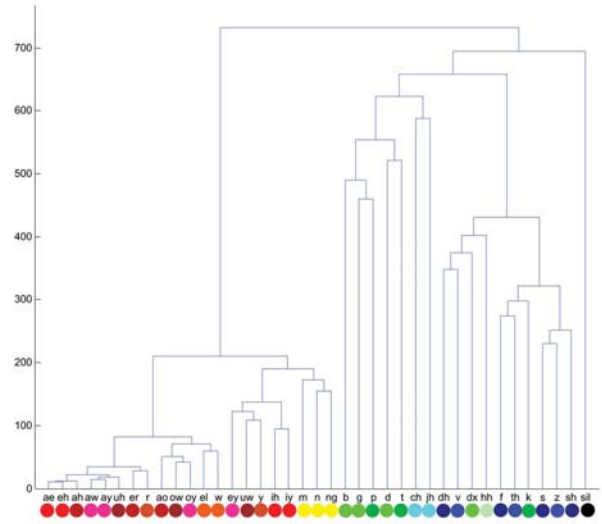


Figure 7. Tree formed by using both views and using the hybrid clustering algorithm. It preserves the separations that were clearly present in each of the individual views.

6. Future Work

We seem to have found a way to effectively exploit multiple views in hierarchical clustering. We still face the issue of how to quantitatively assess the extent of the benefit. A related approach to multiview learning is presented in (Bickel & Scheffer, 2004). They evaluate their clusterings by computing the entropy of the clusters given the true classification. Our approach of using the histogram of inverse goodness-of-split values is more subjective but better fits the unsupervised model.

We also plan to test our hybrid algorithm on different and larger data sets, such as the WebKB data recently provided to us by Steffen Bickel, for which we are most grateful.

References

Banerjee, A., Merugu, S., Dhillon, I., & Ghosh, J. (2004). Clustering with bregman divergences.

Bickel, S., & Scheffer, T. (2004). Multi-view clustering. *Fourth IEEE International Conference on Data Mining*.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*.

Collins, M., & Singer, Y. (1999). Unsupervised models for named entity classification.

Davies, D., & Bouldin, D. (1979). A cluster separation measure. *IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 1, pp. 224-227.*

Dhillon, I., Mallela, S., & Kumar, R. (2003). A divisive information-theoretic feature clustering algorithm for text classi cation.

Lafferty, J., Pietra, S., & Pietra, V. (1997). Statistical learning algorithms based on bregman distances.

Pereira, F. C. N., Tishby, N., & Lee, L. (1993). Distributional clustering of english words. *Meeting of the Association for Computational Linguistics* (pp. 183–190).

Rabiner, L. R., & Juang, B.-H. (1993). *Fundamentals of speech recognition.* PTR Prentice-Hall, Inc.

Zue, V., & Seneff, S. (1988). Transcription and alignment of the timit database.

Active Learning of Features and Labels

Balaji Krishnapuram BALAJI.KRISHNAPURAM@SIEMENS.COM
Siemens Medical Solutions USA, 51 Valley Stream Parkway, Malvern, PA 19355

David Williams DPW@EE.DUKE.EDU
Ya Xue YA.XUE@DUKE.EDU
Lawrence Carin LCARIN@EE.DUKE.EDU
Department of Electrical and Computer Engineering, Duke University, Box 90291, Durham, NC 27708-0291

Mário A. T. Figueiredo MTF@LX.IT.PT
Instituto de Telecomunicações, Instituto Superior Técnico, 1049-001 Lisboa, Portugal

Alexander J. Hartemink AMINK@CS.DUKE.EDU
Department of Computer Science, Duke University, Box 90129, Durham, NC 27708-0129

Abstract

Co-training improves multi-view classifier learning by enforcing internal consistency between the predicted classes of unlabeled objects based on different views (different sets of features for characterizing the same object). In some applications, due to the cost involved in data acquisition, only a subset of features may be obtained for many unlabeled objects. Observing additional features of objects that were earlier incompletely characterized, increases the data available for co-training, hence improving the classification accuracy. This paper addresses the problem of active learning of features: which additional features should be acquired of incompletely characterized objects in order to maximize the accuracy of the learned classifier? Our method, which extends previous techniques for the active learning of labels, is experimentally shown to be effective in a real-life multi-sensor mine detection problem.

1. Motivation

A fundamental assumption in the field of classifier design is that it is costly to acquire labels; after all, if label acquisition were cheap, we would have little need for classifiers because we could simply acquire labels as and when we needed them. But how does the situation change when it is also costly to acquire features? This paper aims to answer this question. We begin with a little more motivation.

In the simplest setting for classifier design, each object has been characterized by a vector of features and a label, as schematically depicted in Figure 1a. Assuming that labels are indeed costly to acquire, we can imagine relaxing this setting so that each object has been characterized by a vector of features, but only a small subset of the objects has been labeled. If we are not permitted to acquire additional labels for the unlabeled data, as shown in Figure 1b, we are in a semi-supervised learning setting (Belkin et al., 2004; Blum & Chawla, 2001; Corduneanu & Jaakkola, 2004; Inoue & Ueda, 2003; Joachims, 1999; Joachims, 2003; Krishnapuram et al., 2004; Nigam et al., 2000; Seeger, 2001; Zhu et al., 2003); on the other hand, if we *are* permitted to label some of the unlabeled data (Figure 1c), we are in an active learning setting (MacKay, 1992; Muslea et al., 2000; Krishnapuram et al., 2004; Tong & Koller, 2001).

Expanding this framework still further, sometimes the objects to be classified can be characterized by vectors of features in multiple independent ways; we will call each of these characterizations a *view*. For example, a web page may be described either using the words it contains or the set of words in the links pointing to it. A person may be identified on the basis of facial features in an image, speech patterns in an audio recording, or characteristic motions in a video. Buried mines may be investigated using radar, sonar, hyper-spectral, or other kinds of physical sensors. Assuming that only a small subset of the objects has been labeled and that no further labels may be acquired (Figure 1d), we are in the setting of the original co-training algorithm of Blum and Mitchell (1998), which has been extended in a number of interesting directions in subsequent work (Brefeld & Scheffer, 2004; Collins & Singer, 1999; Dasgupta et al., 2001; Balcan et al., 2004). In particular, we recently reformulated co-training using a prior in a

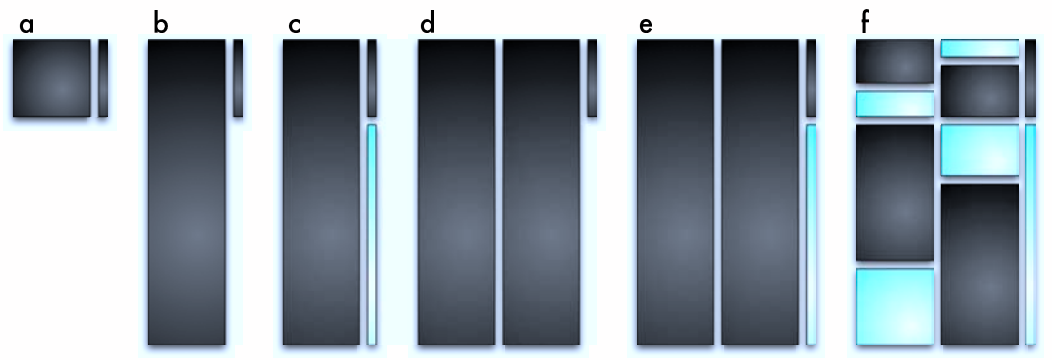


Figure 1. Schematic depiction of different settings. Throughout, rows correspond to objects, wide boxes to feature matrices, and narrow boxes to vectors of class labels; black shading indicates available data, blue shading indicates missing data that can be acquired, and whitespace indicates missing data that cannot be acquired. (a) Each object is characterized by one set of features and one label: supervised learning. (b) Some objects are missing labels that cannot be acquired: semi-supervised learning. (c) Some objects are missing labels that can be acquired: active learning of labels. (d) Objects can be characterized by more than one view, but some are missing labels that cannot be acquired: co-training. (e) Same as (d) but labels can be acquired: active learning of labels with co-training (Krishnapuram et al., 2004). (f) Some objects have not been labeled and not all objects have been characterized in all views: active learning of features and labels (this paper).

Bayesian context (Krishnapuram et al., 2004). This reformulation is based on logistic regression, yielding a convex objective function with a unique local optimum.

As shown in (Krishnapuram et al., 2004), our formulation enables us to consider active learning settings in which we are now permitted to label some of the unlabeled data, as depicted in Figure 1e. But this same formulation also enables us to consider a new setting in which each object may be characterized by only a subset of available views. This can occur in real-life when features are also costly to acquire, as is often the case when physical sensors need to be deployed for each view of an object. If new views may be acquired for any object, as depicted in Figure 1f, how should we decide which view to acquire? And what is the relative benefit of acquiring features versus labels?

In terms of previous work, while several authors have provided criteria for deciding which objects should be labeled (the setting of Figures 1c and 1e), we seek to answer a new question: which incompletely characterized objects (whether labeled or unlabeled) should be further investigated in order to most accurately learn a classifier? To the best of our knowledge, despite its clear importance, the latter question has not been formally addressed before. A few authors have developed intuitive but somewhat *ad hoc* approaches for acquiring features only for labeled objects (Melville et al., 2004; Zheng & Padmanabhan, 2002), but we believe this is the first approach for feature acquisition on both labeled and unlabeled objects.

Section 2 summarizes the probabilistic model for multi-view classifier design that we inherit from Krishnapuram et al. (2004). Section 3 explains the information-theoretic background for the criteria developed in Sections 4 and 5 for active label acquisition and active feature acquisition, respectively. Experimental results are provided in Section 6 and a summary of our conclusions in Section 7.

2. Probabilistic model

2.1. Notation

For notational simplicity, we focus on two-class problems for objects characterized by two views; the proposed methods extend naturally to multi-class and multi-view problems. Since we have only two views, we'll use dot notation to indicate them: let $\dot{\boldsymbol{x}}_i \in \mathbb{R}^{d_1}$ and $\ddot{\boldsymbol{x}}_i \in \mathbb{R}^{d_2}$ be the feature vectors obtained from the two views of the i-th object. Let $\boldsymbol{x}_i = [\dot{\boldsymbol{x}}_i^T, \ddot{\boldsymbol{x}}_i^T]^T$ be the d-dimensional ($d = d_1 + d_2$) vector containing the concatenation of the feature vectors from both views (with appropriate missing values if an object has not been characterized in both views).

In addition to the features in the two views, binary class labels are also collected for a subset of objects; the label of the i-th object is denoted as $y_i \in \{-1, 1\}$. The set of L labeled objects is $\mathcal{D}_L = \{(\boldsymbol{x}_i, y_i) : \boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}\}_{i=1}^{L}$, while the set of U unlabeled objects is $\mathcal{D}_U = \{\boldsymbol{x}_i : \boldsymbol{x}_i \in \mathbb{R}^d\}_{i=L+1}^{L+U}$. Thus, the available training data is $\mathcal{D}_{\text{train}} = \mathcal{D}_L \bigcup \mathcal{D}_U$.

Let $\dot{\mathcal{S}}$, $\ddot{\mathcal{S}}$, and $\dddot{\mathcal{S}} = \dot{\mathcal{S}} \bigcap \ddot{\mathcal{S}}$ denote, respectively, the sets containing the indices of objects characterized by sensor 1, sensor 2, and both. The indices of the corresponding labeled and unlabeled objects are denoted as $\dot{\mathcal{S}}_L$, $\ddot{\mathcal{S}}_L$, $\dddot{\mathcal{S}}_L$

and $\dot{\mathcal{S}}_U, \ddot{\mathcal{S}}_U, \dddot{\mathcal{S}}_U$.

2.2. Multi-view logistic classification

In binary logistic regression, the predicted class probabilities are modeled using the well-known logistic function $\sigma(z) = (1+\exp(-z))^{-1}$. For example, in the first view,

$$P\left(y_i \;\middle|\; \dot{\boldsymbol{x}}_i, \dot{\boldsymbol{w}}\right) = \sigma\left(y_i \dot{\boldsymbol{w}}^T \dot{\boldsymbol{x}}_i\right), \tag{1}$$

where $\dot{\boldsymbol{w}}$ is the classifier weight vector for the first view. A similar expression holds for the second view. Denoting $\boldsymbol{w} = [\dot{\boldsymbol{w}}^T, \ddot{\boldsymbol{w}}^T]^T$, we can find the maximum likelihood (ML) estimate of the classifiers for both sensors $\widehat{\boldsymbol{w}}_{\text{ML}}$, by maximizing the overall log-likelihood,

$$\ell(\boldsymbol{w}) = \ell_1(\dot{\boldsymbol{w}}) + \ell_2(\ddot{\boldsymbol{w}}),$$

where

$$\ell_1(\dot{\boldsymbol{w}}) = \sum_{i \in \dot{\mathcal{S}}_L} \log P\left(y_i \;\middle|\; \dot{\boldsymbol{x}}_i, \dot{\boldsymbol{w}}\right),$$
$$\ell_2(\ddot{\boldsymbol{w}}) = \sum_{i \in \ddot{\mathcal{S}}_L} \log P\left(y_i \;\middle|\; \ddot{\boldsymbol{x}}_i, \ddot{\boldsymbol{w}}\right).$$

Given a prior $p(\boldsymbol{w})$, we can find the maximum *a posteriori* (MAP) estimate $\widehat{\boldsymbol{w}}_{\text{MAP}}$ by maximizing the log-posterior $L(\boldsymbol{w}) = \ell(\boldsymbol{w}) + \log p(\boldsymbol{w})$. Clearly, ML estimation can be accomplished by independently maximizing the log-likelihoods for each sensor, $\ell_1(\dot{\boldsymbol{w}})$ and $\ell_2(\ddot{\boldsymbol{w}})$. If the prior factorizes as $p(\boldsymbol{w}) = p_1(\dot{\boldsymbol{w}})\, p_2(\ddot{\boldsymbol{w}})$ (*i.e.* , it models $\dot{\boldsymbol{w}}$ and $\ddot{\boldsymbol{w}}$ as *a priori* independent) we can clearly still perform MAP estimation of the two classifiers separately. However, if $p(\boldsymbol{w})$ expresses some dependence between $\dot{\boldsymbol{w}}$ and $\ddot{\boldsymbol{w}}$, both classifiers must be trained simultaneously by jointly maximizing $L(\boldsymbol{w})$. In this case, the classifier learned for each sensor also depends on the data from the other sensor. This provides a Bayesian mechanism for sharing information and thus exploiting synergies in learning classifiers for different sensors.

2.3. Co-training priors

The standard means of coordinating information from both sensors is by using the concept of *co-training* (Blum & Mitchell, 1998): on the objects with indices in $\dddot{\mathcal{S}}_U$, the two classifiers should agree as much as possible. In a logistic regression framework, the disagreement between the two classifiers on the objects in $\dddot{\mathcal{S}}$ can be measured by

$$\sum_{i \in \dddot{\mathcal{S}}_U} \left(\dot{\boldsymbol{w}}^T \dot{\boldsymbol{x}}_i - \ddot{\boldsymbol{w}}^T \ddot{\boldsymbol{x}}_i\right)^2 = \boldsymbol{w}^T \mathbf{C}\, \boldsymbol{w}, \tag{2}$$

where $\mathbf{C} = \sum_{i \in \dddot{\mathcal{S}}_U} [\dot{\boldsymbol{x}}_i^T, -\ddot{\boldsymbol{x}}_i^T]^T [\dot{\boldsymbol{x}}_i^T, -\ddot{\boldsymbol{x}}_i^T]$. This suggests the following Gaussian "co-training prior"

$$p(\boldsymbol{w}) = p(\dot{\boldsymbol{w}}, \ddot{\boldsymbol{w}}) \propto \exp\left\{-(\lambda_{co}/2)\, \boldsymbol{w}^T \mathbf{C}\, \boldsymbol{w}\right\}. \tag{3}$$

This co-training prior can be combined with other *a priori* information, also formulated in the form of Gaussian priors, derived from labeled and unlabeled data using the formulation in (Krishnapuram et al., 2004). Formally,

$$p(\boldsymbol{w}|\boldsymbol{\lambda}) = \mathcal{N}\left(\boldsymbol{w} \;\middle|\; \mathbf{0}; \left(\boldsymbol{\Delta}_{\text{prior}}(\boldsymbol{\lambda})\right)^{-1}\right), \tag{4}$$

where the prior precision matrix $\boldsymbol{\Delta}_{\text{prior}}(\boldsymbol{\lambda})$, which is a function of a set of parameters (including λ_{co}) collected in vector $\boldsymbol{\lambda}$, is

$$\boldsymbol{\Delta}_{\text{prior}}(\boldsymbol{\lambda}) = \boldsymbol{\Lambda} + \lambda_{co}\mathbf{C} + \begin{bmatrix} \dot{\lambda}\dot{\boldsymbol{\Delta}} & \mathbf{0} \\ \mathbf{0} & \ddot{\lambda}\ddot{\boldsymbol{\Delta}} \end{bmatrix} \tag{5}$$

with $\boldsymbol{\Lambda} = \text{diag}\{\lambda_1, ..., \lambda_d\}$; finally

$$\dot{\boldsymbol{\Delta}} = \sum_{i,j \in \dot{\mathcal{S}}, i>j} \dot{K}_{ij} \left(\dot{\boldsymbol{x}}_i - \dot{\boldsymbol{x}}_j\right)\left(\dot{\boldsymbol{x}}_i - \dot{\boldsymbol{x}}_j\right)^T$$

is the precision matrix for semi-supervised learning derived in Krishnapuram et al. (2004), and $\ddot{\boldsymbol{\Delta}}$ is a similar expression. All the parameters in $\boldsymbol{\lambda}$ formally play the role of inverse variances; thus, they are given conjugate gamma hyper-priors. If we let $\dot{\lambda} = \ddot{\lambda} = \lambda_0$, then we have:

$$\begin{aligned} p\left(\lambda_0 \;\middle|\; \alpha_0, \beta_0\right) &= \text{Ga}\left(\lambda_0 \;\middle|\; \alpha_0, \beta_0\right), \\ p\left(\lambda_i \;\middle|\; \alpha_1, \beta_1\right) &= \text{Ga}\left(\lambda_i \;\middle|\; \alpha_1, \beta_1\right), \\ p\left(\lambda_{co} \;\middle|\; \alpha_{co}, \beta_{co}\right) &= \text{Ga}\left(\lambda_{co} \;\middle|\; \alpha_{co}, \beta_{co}\right). \end{aligned}$$

Under this formulation, it is possible to interpret $\boldsymbol{\lambda}$ as a hidden variable and write a generalized EM (GEM) algorithm for obtaining an MAP estimate $\widehat{\boldsymbol{w}}_{\text{MAP}}$. It is easy to check that the complete-data log-likelihood is linear with respect to $\boldsymbol{\lambda}$; thus, in each iteration of the GEM algorithm, the E-step reduces to the computation of the conditional expectation of $\boldsymbol{\lambda}$ given the current parameter estimate and the observed data (this can be done analytically due to conjugacy). The (generalized) M-step then consists of maximizing a lower bound on the complete log-likelihood (see (Böhning, 1992)) to obtain the new classifier estimate. The steps are repeated until some convergence criterion is met.

3. Information-theoretic criteria for active data acquisition

This section is devoted to answering the following question: what additional information should be added to $\mathcal{D}_{\text{train}}$ so that the classifier parameters $\boldsymbol{w}$ are learned most accurately, at minimum expense? Observe that there are several ways in which $\mathcal{D}_{\text{train}}$ can be augmented: (1) label information y_i for a previously unlabeled object $\boldsymbol{x}_i \in \mathcal{D}_U$; (2) features from sensor 1 for an *unlabeled* object $i \in \ddot{\mathcal{S}}_U \setminus \dddot{\mathcal{S}}_U$ (*i.e.,* such that sensor 2 has been acquired, but 1 has not);

(3) features from sensor 2 for an *unlabeled* object $i \in \mathcal{S}_U^1 \setminus \dot{\ddot{\mathcal{S}}}_U$; (4) and (5) same as (2) and (3), but for labeled objects. In this section, we show how information-theoretic tools can be used to choose the best object to be queried for further information under each scenario.

3.1. Laplace approximation for the posterior density

Ignoring the hyper-priors on the regularizer $\boldsymbol{\lambda}$ (*i.e.* , assuming a fixed $\boldsymbol{\lambda}$), after estimating a classifier $\widehat{\boldsymbol{w}}_{\text{MAP}}$ from training data $\mathcal{D}_{\text{train}}$, a Laplace approximation models the posterior density $p(\boldsymbol{w}|\mathcal{D}_{\text{train}})$ as a Gaussian

$$p(\boldsymbol{w}|\mathcal{D}_{\text{train}}) \approx \mathcal{N}\left(\boldsymbol{w}|\widehat{\boldsymbol{w}}_{\text{MAP}}; (\boldsymbol{\Delta}_{\text{post}})^{-1}\right). \tag{6}$$

Under the logistic log-likelihood and the Gaussian prior (4) herein considered, the posterior precision matrix of the Laplace approximation is given by:

$$\boldsymbol{\Delta}_{\text{post}} = \boldsymbol{\Delta}_{\text{prior}}(\boldsymbol{\lambda}) + \boldsymbol{\Psi} \tag{7}$$

where $\boldsymbol{\Delta}_{\text{prior}}(\boldsymbol{\lambda})$ is the prior precision matrix in (5) and $\boldsymbol{\Psi} = \text{block-diag}\{\dot{\boldsymbol{\Psi}}, \ddot{\boldsymbol{\Psi}}\}$ is the Hessian of the negative log-likelihood (see, *e.g.* , (Böhning, 1992)) where

$$\dot{\boldsymbol{\Psi}} = \sum_{i \in \dot{\mathcal{S}}_L} \dot{p}_i \left(1 - \dot{p}_i\right) \dot{\boldsymbol{x}}_i \dot{\boldsymbol{x}}_i^T,$$

with $\dot{p}_i = \sigma\left(\dot{\boldsymbol{w}}^T \dot{\boldsymbol{x}}_i\right)$; a similar expression holds for $\ddot{\boldsymbol{\Psi}}$.

The differential entropy of the Gaussian posterior under the Laplace approximation is thus ($|\cdot|$ denotes determinant)

$$h(\boldsymbol{w}) = -\frac{1}{2} \log \frac{|\boldsymbol{\Delta}_{\text{post}}|}{2\pi e}. \tag{8}$$

3.2. Mutual information

After estimating a classifier $\widehat{\boldsymbol{w}}_{\text{MAP}}$ from $\mathcal{D}_{\text{train}}$, the (un)certainty in the label y_i predicted for an unlabeled object $\boldsymbol{x}_i \in \mathcal{D}_U$ is given by the logistic model (1): $P(y_i|\boldsymbol{x}_i, \widehat{\boldsymbol{w}}_{\text{MAP}})$. For a object (labeled or not) for which we have $\dot{\boldsymbol{x}}_i$ but not $\ddot{\boldsymbol{x}}_i$ ($i \in \dot{\mathcal{S}} \setminus \dot{\ddot{\mathcal{S}}}$), the uncertainty in the latter can be modeled by some representation of $p(\ddot{\boldsymbol{x}}_i|\dot{\boldsymbol{x}}_i)$ learned from the training objects in $\dot{\ddot{\mathcal{S}}}$.

The mutual information (MI) between $\boldsymbol{w}$ and y_i is the *expected* decrease in entropy of $\boldsymbol{w}$ when y_i is observed,

$$\begin{aligned} I\left(\boldsymbol{w}; y_i\right) &= h(\boldsymbol{w}) - \mathbb{E}\left[h(\boldsymbol{w}|y_i)\right] \\ &= \mathbb{E}\left[\log \left|\boldsymbol{\Delta}_{\text{post}}^{y_i}\right|\right] - \log \left|\boldsymbol{\Delta}_{\text{post}}\right|, \end{aligned} \tag{9}$$

where the expectation is w.r.t y_i with probability distribution $P(y_i|\boldsymbol{x}_i, \widehat{\boldsymbol{w}}_{\text{MAP}})$, while $\boldsymbol{\Delta}_{\text{post}}^{y_i}$ is the posterior precision matrix of the re-trained classifier after observing y_i.

Similarly, the MI between $\boldsymbol{w}$ and a previously unobserved feature $\ddot{\boldsymbol{x}}_i$ (for $i \in \dot{\mathcal{S}} \setminus \dot{\ddot{\mathcal{S}}}$) is given by

$$\begin{aligned} I\left(\boldsymbol{w}; \ddot{\boldsymbol{x}}_i\right) &= h(\boldsymbol{w}) - \mathbb{E}\left[h(\boldsymbol{w}|\ddot{\boldsymbol{x}}_i)|\dot{\boldsymbol{x}}_i\right] \\ &= \mathbb{E}\left[\log \left|\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}\right| \middle| \dot{\boldsymbol{x}}_i\right] - \log \left|\boldsymbol{\Delta}_{\text{post}}\right|, \end{aligned} \tag{10}$$

where the expectation is over the uncertainty $p(\ddot{\boldsymbol{x}}_i|\dot{\boldsymbol{x}}_i)$ and $\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}$ is the posterior precision matrix of the retrained classifier after seeing features from sensor 2 for object i.

The maximum MI criterion has been used before to identify the "best" unlabeled object for which to obtain an additional label (MacKay, 1992):

$$i^* = \underset{i:\boldsymbol{x}_i \in \mathcal{D}_U}{\arg\max}\, I\left(\boldsymbol{w}; y_i\right) = \underset{i:\boldsymbol{x}_i \in \mathcal{D}_U}{\arg\max}\, \mathbb{E}\left[\log \left|\boldsymbol{\Delta}_{\text{post}}^{y_i}\right|\right]. \tag{11}$$

Based on the same criterion, the best object for which to acquire sensor 2 features—among $\dot{\mathcal{S}} \setminus \ddot{\mathcal{S}}$ for which we have features from sensor 1, but not sensor 2—would be

$$i^\dagger = \underset{i \in \dot{\mathcal{S}} \setminus \dot{\ddot{\mathcal{S}}}}{\arg\max}\, \mathbb{E}\left[\log \left|\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}\right| \middle| \dot{\boldsymbol{x}}_i\right]. \tag{12}$$

3.3. Upper bound on mutual information

Unfortunately, $\mathbb{E}[\log |\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}|]$ is very difficult to compute for our models. Alternatively, we compute an upper bound and use it in the maximum MI criterion just presented. Since the function $\log|\mathbf{X}|$ is concave (Boyd & Vandenberghe, 2003), by Jensen's inequality we obtain

$$\mathbb{E}\left[\log|\mathbf{X}|\right] \leq \log\left|\mathbb{E}\left[\mathbf{X}\right]\right|. \tag{13}$$

Hence, our sample selection criterion will be

$$i^\dagger = \underset{i \in \dot{\mathcal{S}} \setminus \dot{\ddot{\mathcal{S}}}}{\arg\max} \left|\mathbb{E}\left[\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i} \middle| \dot{\boldsymbol{x}}_i\right]\right|, \tag{14}$$

instead of the original (12). Intuitively, we try to maximize the expected posterior precision of the parameters.

3.4. Simplifying assumptions

We make two simplifying assumptions, fundamental in making our approach practical for real-life problems.

Assumption 1: Let the posterior density of the parameters, given the original training data $\mathcal{D}_{\text{train}}$, be $p(\boldsymbol{w}|\mathcal{D}_{\text{train}})$. Consider that we obtain additional features $\ddot{\boldsymbol{x}}_i$, for some $i \in \dot{\mathcal{S}} \setminus \dot{\ddot{\mathcal{S}}}$ and retrain the classifier, obtaining a new posterior $p(\boldsymbol{w}|\mathcal{D}_{\text{train}}, \ddot{\boldsymbol{x}}_i)$. When computing the utility of $\ddot{\boldsymbol{x}}_i$, we assume that the the modes of $p(\boldsymbol{w}|\mathcal{D}_{\text{train}}, \ddot{\boldsymbol{x}}_i)$ and $p(\boldsymbol{w}|\mathcal{D}_{\text{train}})$ coincide, although their precision matrices may not. It turns out that it will be possible to obtain the new precisions, without actually re-training, which would be very computationally expensive. It is important to highlight that,

after a "best" index $i^\dagger$ is chosen (under this simplifying assumption), we actually observe $\ddot{\boldsymbol{x}}_{i^\dagger}$ and re-train the classifier, thus updating the mode of the posterior. Since this re-training is done only once for each additional feature acquisition, tremendous computational effort is saved.

The same assumption is made for label acquisition.

Assumption 2: For the purpose of computing the utility of acquiring some new data (a label or a set of features), we treat $\boldsymbol{\lambda}$ as deterministic, and fixed at the value of its expectation after convergence of the GEM algorithm mentioned in Section 2.3. This value is substituted in (7) to compute the entropy and the mutual information.

4. Acquiring additional labels

For the sake of completeness, we now review the approach in Krishnapuram et al. (2004) for acquiring labels.

According to Assumption 1, the MAP estimate $\widehat{\boldsymbol{w}}_{\text{MAP}}$ does not change when $\mathcal{D}_{\text{train}}$ is augmented with a new label y_i; consequently, the class probability estimates are also unchanged. Based on (7), if we obtain the label y_i, for some $\boldsymbol{x}_i \in \mathcal{D}_U$, regardless of whether $y_i = -1$ or $y_i = 1$, the posterior precision matrix becomes

$$\begin{aligned}\boldsymbol{\Delta}_{\text{post}}^{y_i} = \boldsymbol{\Delta}_{\text{post}} &+ \dot{p}_i\,(1-\dot{p}_i)\begin{bmatrix}\dot{\boldsymbol{x}}_i\\ \mathbf{0}\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{x}}_i\\ \mathbf{0}\end{bmatrix}^T \\ &+ \ddot{p}_i\,(1-\ddot{p}_i)\begin{bmatrix}\mathbf{0}\\ \ddot{\boldsymbol{x}}_i\end{bmatrix}\begin{bmatrix}\mathbf{0}\\ \ddot{\boldsymbol{x}}_i\end{bmatrix}^T\end{aligned} \tag{15}$$

The unlabeled object maximizing $|\boldsymbol{\Delta}_{\text{post}}^{y_i}|$ is thus queried for its label. Intuitively, this favors objects with uncertain class probability estimates ($\dot{p}_i$ and/or $\ddot{p}_i$ close to $1/2$).

5. Acquiring additional features

In this section we show how to compute $\mathbb{E}[\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}|\dot{\boldsymbol{x}}_i]$, which is needed to implement the criterion in (14). Due to symmetry, $\mathbb{E}[\boldsymbol{\Delta}_{\text{post}}^{\dot{\boldsymbol{x}}_i}|\ddot{\boldsymbol{x}}_i]$ is computed in a similar fashion, and hence will not be explicitly described. Two different cases must be studied: when $\boldsymbol{x}_i$ is labeled or unlabeled.

5.1. Additional features for unlabeled objects

Equation (7) shows that if we acquire $\ddot{\boldsymbol{x}}_i$ on a object previously characterized by $\dot{\boldsymbol{x}}_i$, matrix $\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}$ becomes

$$\begin{aligned}\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i} = \boldsymbol{\Delta}_{\text{post}} &+ \ddot{\lambda}\sum_{j\in\ddot{\mathcal{S}}}\ddot{K}_{ij}\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{S}_{ij}\end{bmatrix} \\ &+ \lambda_{co}\begin{bmatrix}\dot{\boldsymbol{x}}_i\dot{\boldsymbol{x}}_i^T & -\dot{\boldsymbol{x}}_i\ddot{\boldsymbol{x}}_i^T\\ -\ddot{\boldsymbol{x}}_i\dot{\boldsymbol{x}}_i^T & \ddot{\boldsymbol{x}}_i\ddot{\boldsymbol{x}}_i^T\end{bmatrix},\end{aligned} \tag{16}$$

where

$$\begin{aligned}\mathbf{S}_{ij} = \ddot{\boldsymbol{x}}_i\ddot{\boldsymbol{x}}_i^T - \ddot{\boldsymbol{x}}_i\ddot{\boldsymbol{x}}_j^T \\ - \ddot{\boldsymbol{x}}_j\ddot{\boldsymbol{x}}_i^T + \ddot{\boldsymbol{x}}_j\ddot{\boldsymbol{x}}_j^T.\end{aligned} \tag{17}$$

To compute the conditional expectation $\mathbb{E}[\boldsymbol{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}|\dot{\boldsymbol{x}}_i]$ (see (14)) we need a model for $p(\ddot{\boldsymbol{x}}_i|\dot{\boldsymbol{x}}_i)$. To this end, we use a Gaussian mixture model (GMM) to represent the joint density:

$$p(\ddot{\boldsymbol{x}}_i, \dot{\boldsymbol{x}}_i) = \sum_c \pi_c \mathcal{N}\left(\dot{\boldsymbol{x}}|\dot{\boldsymbol{\mu}}_c, \dot{\boldsymbol{\Sigma}}_c\right)\mathcal{N}\left(\ddot{\boldsymbol{x}}|\ddot{\boldsymbol{\mu}}_c, \ddot{\boldsymbol{\Sigma}}_c\right).$$

Notice that, although using component-wise independence, this joint GMM globally models the dependency between $\dot{\boldsymbol{x}}$ and $\ddot{\boldsymbol{x}}$. From this joint GMM, it is straightforward to derive the conditional $p(\ddot{\boldsymbol{x}}_i|\dot{\boldsymbol{x}}_i)$, which is also a GMM, with weights that depend on $\dot{\boldsymbol{x}}$:

$$p(\ddot{\boldsymbol{x}}|\dot{\boldsymbol{x}}) = \sum_c \pi_c'(\dot{\boldsymbol{x}})\,\mathcal{N}\left(\ddot{\boldsymbol{x}}|\ddot{\boldsymbol{\mu}}_c, \ddot{\boldsymbol{\Sigma}}_c\right). \tag{18}$$

Further, the $\dot{K}_{ij}$ and $\ddot{K}_{ij}$ are set to Gaussian kernels; *e.g.* ,

$$\dot{K}_{ij} = \mathcal{N}\left(\dot{\boldsymbol{x}}_i|\dot{\boldsymbol{x}}_j, \boldsymbol{\Sigma}_\kappa\right). \tag{19}$$

Using (18), (19) and standard Gaussian identities, the required expectations are obtained analytically:

$$\mathbb{E}\left[\ddot{\boldsymbol{x}}_i|\dot{\boldsymbol{x}}_i\right] = \sum_c \pi_c'(\dot{\boldsymbol{x}}_i)\;\ddot{\boldsymbol{\mu}}_c = \boldsymbol{m}_1$$

$$\begin{aligned}\mathbb{E}\left[\ddot{\boldsymbol{x}}_i\ddot{\boldsymbol{x}}_i^T\middle|\,\dot{\boldsymbol{x}}_i\right] \\ = \sum_c \pi_c'(\dot{\boldsymbol{x}}_i)\left(\ddot{\boldsymbol{\mu}}_c\,\ddot{\boldsymbol{\mu}}_c^T + \ddot{\boldsymbol{\Sigma}}_c\right) = \mathbf{M}_2\end{aligned}$$

$$\mathbb{E}\left[\ddot{K}_{ij}\middle|\,\dot{\boldsymbol{x}}_i\right] = \sum_c \pi_c'(\dot{\boldsymbol{x}}_i)\;z_{cj} = m_{3j}$$

$$\mathbb{E}\left[\ddot{K}_{ij}\ddot{\boldsymbol{x}}_i\middle|\,\dot{\boldsymbol{x}}_i\right] = \sum_c \pi_c'(\dot{\boldsymbol{x}}_i)\;z_{cj}\;\boldsymbol{\mu}_{cj} = \boldsymbol{m}_{4j}$$

$$\begin{aligned}\mathbb{E}\left[\ddot{K}_{ij}\ddot{\boldsymbol{x}}_i\ddot{\boldsymbol{x}}_i^T\middle|\,\dot{\boldsymbol{x}}_i\right] \\ = \sum_c \pi_c'(\dot{\boldsymbol{x}}_i)\;z_{cj}\;\left(\ddot{\boldsymbol{\mu}}_{cj}\ddot{\boldsymbol{\mu}}_{cj}^T + \boldsymbol{\Lambda}_c\right) = \mathbf{M}_{5j}.\end{aligned}$$

where

$$\begin{aligned}\boldsymbol{\Lambda}_c &= \left(\ddot{\boldsymbol{\Sigma}}_c^{-1} + \boldsymbol{\Sigma}_\kappa^{-1}\right)^{-1} \\ \boldsymbol{\mu}_{cj} &= \boldsymbol{\Lambda}_c\left(\ddot{\boldsymbol{\Sigma}}_c^{-1}\ddot{\boldsymbol{\mu}}_c + \boldsymbol{\Sigma}_\kappa^{-1}\ddot{\boldsymbol{x}}_j\right)\end{aligned}$$

and

$$z_{cj} = (2\pi)^{-d/2} \, |\mathbf{\Lambda}_c|^{1/2} |\ddot{\mathbf{\Sigma}}_c|^{-1/2} |\mathbf{\Sigma}_\kappa|^{-1/2} \exp\left\{ -\frac{\ddot{\boldsymbol{\mu}}_c^T \ddot{\mathbf{\Sigma}}_c^{-1} \ddot{\boldsymbol{\mu}}_c + \ddot{\boldsymbol{x}}_j^T \mathbf{\Sigma}_\kappa^{-1} \ddot{\boldsymbol{x}}_j - \boldsymbol{\mu}_{cj}^T \mathbf{\Lambda}_c^{-1} \boldsymbol{\mu}_{cj}}{2} \right\}.$$

Finally,

$$\mathbb{E}\left[\mathbf{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i} \middle| \dot{\boldsymbol{x}}_i \right] = \mathbf{\Delta}_{\text{post}} + \ddot{\lambda} \sum_{j \in \ddot{\mathcal{S}}} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{ij} \end{bmatrix} + \lambda_{co} \begin{bmatrix} \dot{\boldsymbol{x}}_i \dot{\boldsymbol{x}}_i^T & -\dot{\boldsymbol{x}}_i \boldsymbol{m}_1^T \\ -\boldsymbol{m}_1 \dot{\boldsymbol{x}}_i^T & \mathbf{M}_2 \end{bmatrix}, \quad (20)$$

where

$$\mathbf{S}_{ij} = m_{3j} \ddot{\boldsymbol{x}}_j \ddot{\boldsymbol{x}}_j^T - \ddot{\boldsymbol{x}}_j \boldsymbol{m}_{4j}^T - \boldsymbol{m}_{4j} \ddot{\boldsymbol{x}}_j^T + \mathbf{M}_{5j}.$$

Substituting (20) into (14) gives us our selection criterion.

5.2. Additional features for labeled objects

From (7), we can derive $\mathbf{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}$ for the case when $\boldsymbol{x}_i$ is a labeled object ($\boldsymbol{x}_i \in \mathcal{D}_L$):

$$\mathbf{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i} = \mathbf{\Delta}_{\text{post}} + \ddot{p}_i (1 - \ddot{p}_i) \begin{bmatrix} \mathbf{0} \\ \ddot{\boldsymbol{x}}_i \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \ddot{\boldsymbol{x}}_i \end{bmatrix}^T + \lambda_{co} \begin{bmatrix} \dot{\boldsymbol{x}}_i \dot{\boldsymbol{x}}_i^T & -\dot{\boldsymbol{x}}_i \ddot{\boldsymbol{x}}_i^T \\ -\ddot{\boldsymbol{x}}_i \dot{\boldsymbol{x}}_i^T & \ddot{\boldsymbol{x}}_i \ddot{\boldsymbol{x}}_i^T \end{bmatrix} + \ddot{\lambda} \sum_{j \in \ddot{\mathcal{S}}} \ddot{K}_{ij} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{ij} \end{bmatrix}. \quad (21)$$

Using standard Gaussian identities and the approximation

$$\sigma(z)\sigma(-z) \approx \mathcal{N}\left(z \,\middle|\, 0, \frac{8}{\pi} \right),$$

we can show that,

$$\mathbb{E}\left[\ddot{p}_i (1 - \ddot{p}_i) \, \ddot{\boldsymbol{x}}_i \ddot{\boldsymbol{x}}_i^T \middle| \dot{\boldsymbol{x}}_i \right] = \sum_c \pi_c'(\dot{\boldsymbol{x}}_i) \left(\boldsymbol{u}_c \boldsymbol{u}_c^T + \mathbf{U}_c \right) l_c = \mathbf{M}_6, \quad (22)$$

where

$$l_c = \mathcal{N}\left(\ddot{\boldsymbol{w}}^T \ddot{\boldsymbol{\mu}}_c \,\middle|\, 0, \frac{8}{\pi} + \ddot{\boldsymbol{w}}^T \ddot{\mathbf{\Sigma}}_c \ddot{\boldsymbol{w}} \right),$$

$$\mathbf{U}_c = \left(\ddot{\mathbf{\Sigma}}_c^{-1} + \frac{\pi}{8} \, \ddot{\boldsymbol{w}} \ddot{\boldsymbol{w}}^T \right)^{-1},$$

and $\boldsymbol{u}_c = \mathbf{U}_c \, \ddot{\mathbf{\Sigma}}_c^{-1} \ddot{\boldsymbol{\mu}}_c$. Finally, we can compute

$$\mathbb{E}\left[\mathbf{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i} \middle| \dot{\boldsymbol{x}}_i \right] = \mathbf{\Delta}_{\text{post}} + \ddot{\lambda} \sum_{j \in \ddot{\mathcal{S}}} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{ij} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_6 \end{bmatrix} + \lambda_{co} \begin{bmatrix} \dot{\boldsymbol{x}}_i \dot{\boldsymbol{x}}_i^T & -\dot{\boldsymbol{x}}_i \boldsymbol{m}_1^T \\ -\boldsymbol{m}_1 \dot{\boldsymbol{x}}_i^T & \mathbf{M}_2 \end{bmatrix}$$

and substitute it into (14).

5.3. Sample requirements and practical approximations

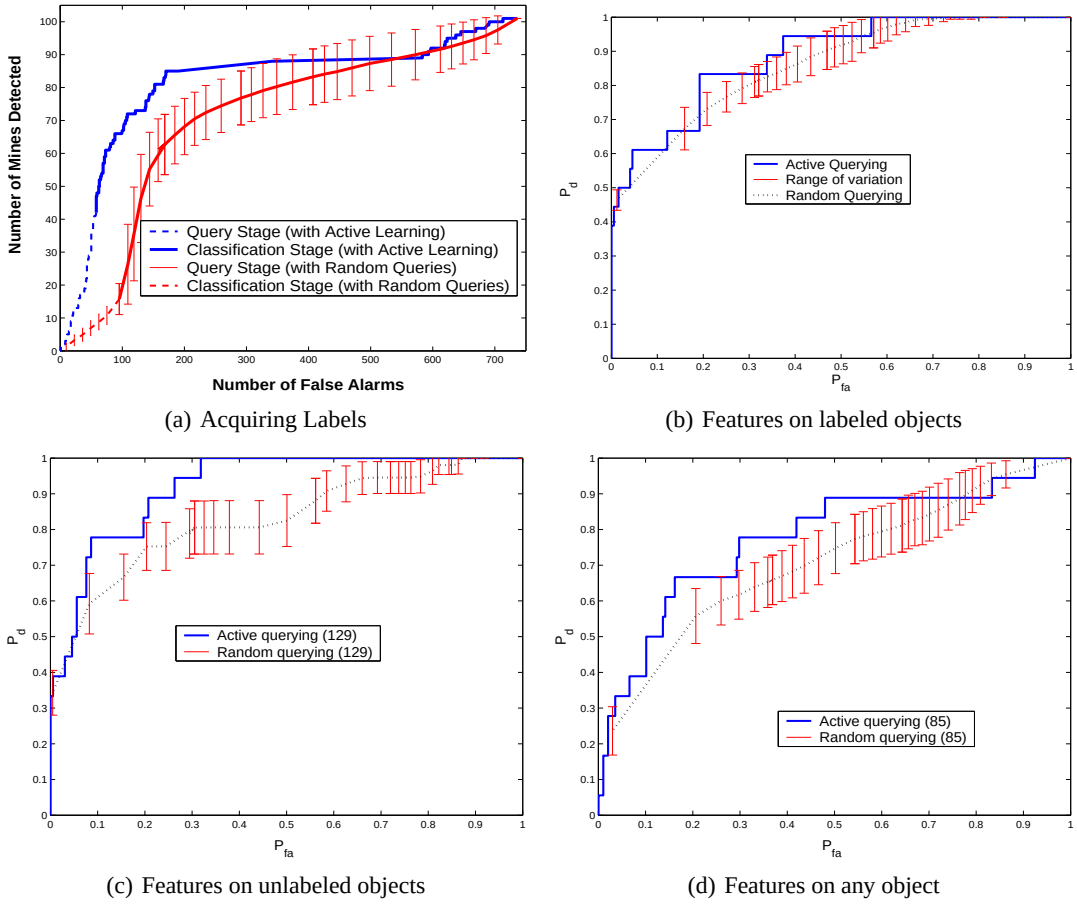
The conditional distribution (18) used to compute $\mathbb{E}[\mathbf{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i} | \dot{\boldsymbol{x}}_i]$ in Sections 5.1 and 5.2 relies on a Gaussian mixture model (GMM) for $p(\ddot{\boldsymbol{x}}_i, \dot{\boldsymbol{x}}_i)$. Unfortunately, fitting an accurate GMM demands a large number of samples; *i.e.* , $\dot{\ddot{\mathcal{S}}}$ must be large relative to $d_1 + d_2$. While our (unreported) studies on simulated data confirmed that the statistical methods proposed above work well when a sufficient number of samples is already available in $\dot{\ddot{\mathcal{S}}}$, in many real-life problems each sensor provides a large number of features, and the above requirement may not be satisfied (especially in early stages of the active learning process). The estimation of covariances is particularly problematic in these small-sample cases.

Due to this difficulty, in the results presented in the next section we use an alternative surrogate for $\mathbb{E}[\mathbf{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}]$. Specifically, in the formulae for $\mathbf{\Delta}_{\text{post}}^{\ddot{\boldsymbol{x}}_i}$ ((16) and (21)) we simply replace $\ddot{\boldsymbol{x}}_i$ with $\boldsymbol{m}_1 = \mathbb{E}[\ddot{\boldsymbol{x}}_i | \dot{\boldsymbol{x}}_i]$—which can still be reliably estimated from limited data, since it does not involve covariances—and subsequently compute the determinant of the resulting matrix. As demonstrated in the next section, this approximation still yields very good experimental results as compared to the random acquisition of additional features.

6. Experiments: Multi-view feature acquisition vs. label acquisition

To evaluate the methods proposed in this paper, we use the same data used in Krishnapuram et al. (2004) to study the performance of co-training and active label acquisition algorithms. Mirroring their experimental setup, we also operate our algorithms transductively, testing the accuracy of the classifier on the same unlabeled data used for semi-supervised training. In brief, the goal was to detect surface and subsurface land mines, using two sensors: (1) a 70-band hyper-spectral electro-optic (EOIR) sensor which provides 420 features; and (2) an X-band synthetic aperture radar (SAR) which provides 9 features. Our choice of dataset was influenced by two factors: lack of other publicly available multi-sensor datasets; a need to compare the benefits of the proposed active feature acquisition strategy against the benefits of adaptive label querying methods.

The results for active feature acquisition on the unlabeled samples (Section 5.1), and on the labeled samples (Section 5.2) are shown in Figure 2. Additionally we let the algorithm automatically decide whether to query additional features on labeled or unlabeled data at each iteration, based on the bound on mutual information for the best candidate query in each case. The results for this are also pro-



(a) Acquiring Labels

(b) Features on labeled objects

(c) Features on unlabeled objects

(d) Features on any object

Figure 2. Multi-sensor adaptive data acquisition with EOIR and SAR features. (a) (dotted) Number of land mines detected during the querying for 100 labels (solid) ROC for the remaining objects. Reproduced from Krishnapuram et al. (2004). (b) ROC after acquiring 27 additional feature sets for incompletely characterized labeled objects. (c) ROC after acquiring 129 additional feature sets for incompletely characterized unlabeled objects. (d) ROC after acquiring 85 features for either labeled or unlabeled objects. Error bars represent one s.d. from the mean.

vided in Figure 2. In all cases, for a baseline comparison, we also provide average ROCs for 100 trials with random querying, with error bars representing one standard deviation from the mean. For additional insight, we also reproduce the results from Krishnapuram et al. (2004) for active label query selection (Section 4) on the same data.

Analysis of results: all the adaptive data acquisition algorithms show significant benefits over the baseline random methods. Nevertheless, as compared to random sample query selection, active learning exhibits maximum additional benefits in two scenarios: label query selection and additional feature acquisition on the unlabeled samples.

Since labeled data is more valuable than unlabeled data, the intelligent choice of a small set of additional label queries improves the classifier performance most. The acquisition of additional features on the unlabeled data also serves to disambiguate the most doubtful test objects, in addition to improving the classifier itself. Since the labeled data do not need further disambiguation, we expect active acquisition of features for labeled objects to exhibit a smaller (but still statistically significant) improvement in accuracy, especially in a transductive experimental setting. We have verified these intuitions by experimentally querying a varying number of objects in each case, although we present only one example result in Figure 2.

7. Conclusions

Using simple but practical approximations, this paper relies on an information-theoretic criterion to answer the question: Which feature sensor should be used to make measurements on objects in order to accurately design multi-sensor classifiers? Since a sensor may be used to obtain

more than one feature simultaneously, this is a more general problem than that of greedily choosing which feature must be obtained in a myopic way, although it subsumes the latter problem as a special case (especially in supervised settings when co-training effects are ignored by fixing $\lambda_{co} = 0$). Despite the potentially wide applicability, we have not seen this question addressed systematically in the literature. Results on measured data indicate that the proposed criterion for adaptive characterization of unlabeled objects significantly improves classifier accuracy; results using the corresponding criterion for labeled objects are less impressive though.

In learning a classifier, one attempts to minimize the error rate on an infinite set of future test samples drawn from the underlying data-generating distribution. However, in transductive settings, one may sometimes only care about classifying the unlabeled training samples. Future work includes extensions of the ideas proposed here to automatically select the sensor whose deployment will most improve the accuracy on *the remaining unlabeled training samples*, instead of attempting to learn accurate classifiers.

We will also consider non-myopic active learning strategies that evaluate the benefits of improved classification accuracy in a setting that explicitly considers both the cost of obtaining class labels and the costs involved in using various sensors to make feature measurements. This would allow us to automatically decide which of the following is the best course of action in any situation: (a) obtain many individually less effective feature measurements (with regard to improving the classification accuracy) using a cheap sensor; or (b) obtain fewer, but more useful feature measurements using an alternative, costlier sensor; or (c) obtain a small number of additional class labels at a significant cost.

References

Balcan, M.-F., Blum, A., & Yang, K. (2004). Co-training and expansion: Towards bridging theory and practice. *NIPS*. Cambridge, MA: MIT Press.

Belkin, M., Niyogi, P., & Sindhwani, V. (2004). *Manifold learning: a geometric framework for learning from examples* (Technical Report). Dept. Computer Science, U. of Chicago.

Blum, A., & Chawla, S. (2001). Learning from labeled and unlabeled data using graph mincuts. *Proceedings of the 18th International Conference on Machine Learning*.

Blum, A., & Mitchell, T. (1998). Combining labelled and unlabelled data with co-training. *Proc. Eleventh Annual Conference on Computational Learning Theory (COLT) 1998*.

Böhning, D. (1992). Multinomial logisitic regression algorithm. *Annals of the Institute of Statistical Mathematics, 44*, 197–200.

Boyd, S., & Vandenberghe, L. (2003). *Convex Optimization*. Cambridge University Press.

Brefeld, U., & Scheffer, T. (2004). Co-EM support vector learning. *Proceedings of the Twenty-First International Conference on Machine Learning – ICML*.

Collins, M., & Singer, Y. (1999). Unsupervised models for named entity classification. *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Corduneanu, A., & Jaakkola, T. (2004). Distributed information regularization on graphs. *Advances in Neural Information Processing Systems*. Cambridge, MA: MIT Press.

Dasgupta, S., Littman, M., & McAllester, D. (2001). Pac generalization bounds for co-training. *Proc. Neural Info. Processing Systems NIPS*.

Inoue, M., & Ueda, N. (2003). Exploitation of unlabelled sequences in hidden markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 25*, 1570–1581.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. *Proceedings of the 16th International Conference on Machine Learning* (pp. 200–209). San Francisco: Morgan Kaufmann.

Joachims, T. (2003). Transductive learning via spectral graph partitioning. *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*.

Krishnapuram, B., Williams, D., Xue, Y., Hartemink, A., Carin, L., & Figueiredo, M. (2004). On semi-supervised classification. *Advances in Neural Information Processing Systems*. Cambridge, MA: MIT Press.

MacKay, D. J. C. (1992). Information-based objective functions for active data selection. *Neural Computation, 4*, 589–603.

Melville, P., Saar-Tsechansky, M., Provost, F., & Mooney, R. J. (2004). Active feature acquisition for classifier induction. *Proceedings of the Fourth International Conference on Data Mining (ICDM-2004)* (p. (to appear)).

Muslea, I., Minton, S., & Knoblock, C. (2000). Selective sampling with redundant views. *Proc. of National Conference on Artifical Intelligence* (pp. 621–626).

Nigam, K., McCallum, A., Thrun, S., & Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning Journal, 39*, 103–134.

Seeger, M. (2001). *Learning with labelled and unlabelled data* (Technical Report). Institute for Adaptive and Neural Computation, University of Edinburgh, Edinburgh, UK.

Tong, S., & Koller, D. (2001). Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research, 2*, 45–66.

Zheng, Z., & Padmanabhan, B. (2002). On active learning for data acquisition. *Proceedings of the Second International Conference on Data Mining (ICDM-2002)*.

Zhu, X., Lafferty, J., & Ghahramani, Z. (2003). *Semi-supervised learning: From Gaussian fields to Gaussian processes* (Technical Report CMU-CS-03-175). School of CS, CMU.

Multiple Views in Ensembles of Nearest Neighbor Classifiers

Oleg Okun OLEG@EE.OULU.FI
Machine Vision Group, P.O. Box 4500, 90014 University of Oulu, Finland

Helen Priisalu
Tallinn University of Technology, Estonia

Abstract

Multi-view classification is a machine learning methodology when patterns or objects of interest are represented by a set of different views (sets of features) rather than the union of all views. In this paper, multiple views are employed in ensembles of nearest neighbor classifiers where they demonstrate promising results in classifying a challenging data set of protein folds. In particular, up to 4.68% increase in accuracy can be achieved, compared to the best result in single-view classification, thus rendering ensembles of nearest neighbor classifiers employing multiple views an attractive research direction.

1. Introduction

The idea of employing multiple classifiers instead of a single (best) classifier gained significant popularity during last years. Among various strategies is learning with multiple views (feature sets), which implies that each pattern is described by several feature sets rather than their combination into a single set. Yarowsky (1995) and Blum and Mitchell (1998) were first who have noticed that multiple views can lead to better classification accuracy than the union of all views. However, for this to happen, many unlabeled patterns must be available for learning in addition to labeled ones, which ideally fits to the case of semi-supervised classification. Abney (2002) explained the success of multi-view learning methods by the fact that given certain independence between individual views (and therefore base learners), the disagreement between base learners can be minimized by using the unlabeled data, which, in turn, improves the combined accuracy of these learners. Multi-view learning is not, however, limited to semi-supervised classification: for example, several authors (Bickel & Scheffer, 2004; Kailing et al., 2004) applied it for unsupervised clustering.

A fully supervised mode of multi-view learning (Rüping, 2005; Tsochantaridis & Hofmann, 2002) has been not so intensively explored as other modes. To contribute to this research, we propose to employ multiple classifiers or ensembles of classifiers, each making predictions based on a specific view (set(s) of features).

The goal of combining classifiers is to improve accuracy, compared to a single classifier. An ensemble means combining multiple versions of a single classifier through voting. Each version is called an individual classifier. An ensemble of classifiers must be both diverse and accurate in order to improve accuracy, compared to a single classifier. Diversity guarantees that all the individual classifiers do not make the same errors. If the classifiers make identical errors, these errors will propagate to the whole ensemble and hence no accuracy gain can be achieved in combining classifiers. In addition to diversity, accuracy of individual classifiers is important, since too many poor classifiers can overwhelm correct predictions of good classifiers.

In order to make individual classifiers diverse, many ensemble generation methods use feature selection so that each classifier works with a specific feature set. Feature selection can be often time-consuming and in certain cases, almost all features may be relevant so that none of them can be preferred to others. Our hypothesis is that since each view constitutes a set or sets of features, multiple views can be used instead of feature selection in order to achieve diversity while saving time.

Our idea is applied to a challenging bioinformatics task: protein fold recognition. The main challenge

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

comes from the fact that training and test data have low similarity within each set and between sets. In bioinformatics, low similarity between training and test data is a prerequisite for unbiased test error estimation. In addition, many folds are insufficiently represented in both sets: there are often as few as seven representatives per fold! Nevertheless, experiments demonstrate that the accuracy rate can grow by 4.68% when utilizing nine views built from the six original feature sets.

One of the first problems to solve when dealing with ensembles of classifiers is to select a base classifier. In this work, we adapted a variant of the nearest neighbor (NN) classifier (Vincent & Bengio, 2002) because it showed a competitive performance to support vector machines. In next two sections, two popular methods for generating ensembles of classifiers, namely, bagging and boosting, are described together with a brief survey of the known ensembles of NN classifiers.

2. Bagging and Boosting

According to Breiman (1996a), bagging (**b**ootstrap **agg**regat**ing**) is a method for generating multiple versions of a classifier and using them to obtain an aggregated classifier. The multiple versions are generated by making bootstrap (random sampling with replacement) samples of the original training set and considering these samples as new training sets. Each bootstrap sample has the same number of patterns as the original training set. The aggregation is combining predictions (class labels) of the individual versions by majority voting, where all votes are equally weighted. However, bagging will not help if the accuracy of an indivudual classifier is close to the limits attainable on a given data set.

Boosting (Freund & Schapire, 1996) operates by iteratively running a given individual classifier on various distributions over the training data, and then combining predictions of the individual classifiers into a single composite classifier by a weighted vote. Unlike bagging, boosting forces the individual classifier C_{t+1} at the $t+1$th iteration to concentrate on the training patterns misclassified by the individual classifier C_t at the tth iteration through reweighting (assigning larger weight) misclassified patterns. That is, another distinction between boosting and bagging is that the voting process in boosting is not equally weighted in contrast to that of bagging.

Both bagging and boosting perform best for unstable classifiers whose predictions are very sensitive to (small) changes in the training data. Examples of such classifiers are decision trees and neural networks. Unstable classifiers can have low bias but high variance. It implies that stable classifiers like NN can have high bias and low variance. Reducing either bias or variance, or both is the way to lower the test set misclassification error. Both bagging and boosting can reduce the variance while boosting can also lower the bias, even if the variance of a classifier is low (Schapire et al., 1997).

3. Ensembles of NN Classifiers

There is not much work previously done regarding ensembles of NN classifiers. Breiman (1996a) was perhaps one of the first researchers who studied this problem. He concluded that bagging NN classifiers does not lead to increased accuracy because the NN classifier is stable, i.e., errors made by individual classifiers are highly correlated. In fact, bagging stable classifiers can sometimes lower accuracy. The reason of failure for bagging NN classifiers can be that bagging is intended to reduce the variance of a classifier and the variance of the (stable) NN classifier seems to be low so it is hard to decrease it further. Applying bagging to lower the bias is not meant (Freund & Schapire, 1996). However, selecting a subset from the whole set of the original features typically causes instability in classifier predictions: hence bagging may work.

Approaching from another direction, Alpaydin (1997), Bao and Ishii (2002), Oza and Tumer (2001), and Skalak (1996) showed that an ensemble of NN classifiers each of which is trained from the small number of prototypes selected from the whole training set results in better accuracy than a single NN classifier using all training patterns. Alpaydin (1997) argued that NN classifiers do not tend to generate diverse (uncorrelated) votes on large training sets. That is why an ensemble of NN classifiers will not succeed on such data sets. Hence a large set should be either partitioned into m smaller subsets, given m classifiers, or if the training set is not large enough to allow partitioning, one can use bootstrap in order to get smaller data sets from the original set. Alpaydin (1997) concluded that an ensemble of classifiers is superior over a single classifier only if individual classifiers of the ensemble fail under different circumstances, and this requirement is satisfied for small training sets. Oza and Tumer (2001) added that feature selection (they used input decimation) aims at reducing the correlations among individual classifiers by using different subsets of the original features, while bagging and boosting approach to the same goal by choosing different subsets of training patterns. It implies that two approaches are clearly

orthogonal and can complement each other, i.e., feature selection can be incorporated into bagging or/and boosting. Bao and Ishii (2002) applied rough sets for feature selection prior to combining NN classifiers.

Freund and Schapire (1996) combined AdaBoost and a variant of the NN classifier. However, the goal was to speed up classification rather than to achieve higher accuracy. Thus, like in (Alpaydin, 1997; Skalak, 1996), boosting acted like NN editing, which reduces the number of training patterns, leaving only those which are sufficient to correctly label the whole training set. In (Freund & Schapire, 1996) a random subset of training patterns, chosen according to the distribution provided by the boosting algorithm, was used as prototypes, and the standard NN classifier predicted the class label of a test pattern according to its closest prototype. Freund and Schapire (1996) remarked that when the *distributions of the training and test data are different*, boosting can lose its advantage over other methods. In addition, in case of small data sets, patterns (typically outliers) that are consistently misclassified at every iteration, may warp classifiers (Breiman, 1996b) and therefore boosting. However, except these two cases, there is no reason to believe that stability of a classifier per se should lead to failure of boosting (Schapire et al., 1997), though no examples involving boosting and NN classifiers and confirming this statement were provided. Bay (1999) pointed to the following reasons why boosting cannot be useful for ensembles of NN classifiers: 1) boosting stops when a classifier achieves 100% accuracy on the training set, and this always happens for the NN classifier, 2) increasing the weight of a hard to classify pattern does not help to correctly classify that pattern since each pattern can only help in classifying its neighbors, but not itself. O'Sullivan et al. (2000) tried to adapt AdaBoost to ensembles of NN classifiers when the test data significantly differ from the training data because features in the test set are either missing or corrupted. Their approach is called FeatureBoost and it represents a variant of boosting where features are boosted rather than patterns. FeatureBoost conducts a sensitivity analysis on the features used by previously learned models, and then biasing future learning away from the features used most. Though O'Sullivan et al. (2000) reached accuracy improvements over AdaBoost, they indicated that these gains were less striking in the ensembles of NNs, compared to the ensembles of decision trees.

Bay (1998) proposed to use simple (majority) voting in order to combine outputs from multiple NN classifiers, each having access only to a random subset of the original features. Each NN classifier employes the same number of features. Each time a test pattern is presented for classification, a new random subset of features for each classifier is selected. Predictions of the NN classifiers are extremely sensitive to the features used. Hence according to (Bay, 1998), different feature sets lead to diverse individual NN classifiers, making uncorrelated errors in the ensemble. Because of uncorrelated errors simple voting is able to result in the high overall accuracy even though individual classifiers may be not very accurate. Bay (1999) claims that his algorithm is able to reduce both bias and variance. However, he remarked that there is no guarantee that using different feature sets will always decorrelate errors.

A similar conclusion that diversity among selected features decorrelates errors made by individual classifiers, was obtained in (Ricci & Aha, 1998) when applying error-correcting output coding (ECOC) to combine NN classifiers. This work emphasizes importance of *appropriate* feature selection in making errors uncorrelated, which implies that feature selection incorporated into ECOC does not always necessarily produce desirable improvement in accuracy. Although it seems that ECOC can sometimes reduce both variance and bias, nevertheless ECOC needs to convert the k-way classification problem into a set of binary problems.

Tsymbal (2002) proposed dynamic integration of classifiers by arguing that simple majority voting does not take into account the fact that each individual classifier performs best in certain cases, i.e., its region of expertise is localized. If a combining algorithm is able to identify such regions, the accuracy of an ensemble can be significantly improved. The basic idea of dynamic integration is that in addition to training patterns, training errors made by each classifier can be utilized as well. Dynamic intergation estimates the local accuracy of the individual classifiers by analyzing their accuracy on nearby patterns to the pattern to be classified. To ensure that individual classifiers are diverse, Tsymbal (2002) experimented with different distance functions and feature sets chosen by feature selection.

Alkoot and Kittler (2002a; 2002b) concentrated on such methods for combining classifiers as the product rule, which combines multiple classifier outputs (typically class a posteriori probability estimates) by multiplication. The product rule plays a prominent role because of its theoretical basis in probability calculus. Nevertheless, its performance is degraded by the veto effect when at least one of the individual classifiers produces the class a posteriori probability estimate that is equal to zero. In this case, the output of an en-

semble will be zero, too, even though other classifiers can provide a lot of support for the class. In (Alkoot & Kittler, 2002b), Modified product was proposed to mitigate the veto effect. According to this rule, if the output (class a posteriori probability estimate) of a classifier falls below a specified threshold t, it is set to t, while classifier estimates for other classes stay unchanged. Though Modified product outperforms the product rule for certain thresholds, it was nevertheless heuristic. That is why Alkoot and Kittler (2002a) went further in battling the veto effect and marginalized the k-NN estimates using the Bayesian prior (they called it moderation). As a result, the class a posteriori estimate κ/k, where κ is the number of training patterns belonging to a certain class out of k nearest neighbors, is replaced with $(\kappa+1)/(k+m)$, given m classes. Thus, even if $k=1$ and $\kappa=0$, the estimate is $1/(1+m)$ instead of zero. As k increases, the smallest estimate assumes zero only when $k \to \infty$. One advantage of moderating NN classifiers is that bagging with moderation can largely compensate for the lack of training data, i.e., this new rule is effective even for small training sets.

Bao et al. (2004) used simple (majority) voting combining outputs from multiple NN classifiers, where each classifier has access only to a certain distance function. Thus, they imitated the approach (Bay, 1998; Bay, 1999) while replacing different feature sets with different distance functions. Zhou and Yu (2005a) adapted bagging to the NN classifiers by perturbing both training sets and distance functions. As a result, a specific bootstrap sample and distance function are associated with each individual classifier. As in bagging, the combining rule is majority voting. Zhou and Yu (2005a) concluded that neither perturbing training data nor distance functions is effective in building ensembles of NN classifiers based on bagging, but their combination is. Following this line, Zhou and Yu (2005b) utilized multimodal perturbation, i.e., perturbing the training data, features, and distance functions.

Paik and Yang (2004) hypothesize that combining classifiers is not always better than selecting a single (best) classifier. They investigated the ensembles of NN classifiers for tumor classification based on gene expression data and examine relationships between cross-validation (CV) selection instability and ensemble performance. In practice, CV has been used for two different purposes: selecting the best model among a set of models and finding the optimal parameters of a given model. Similarly to other model selection techniques, CV has to face the problem of uncertainty of selection when there are several almost identically performing models. In such a case, the uncertainty of selection is high and a slight change of the data can dramatically affect the classifier performance, i.e., under these circumstances selecting the best classifier is ambiguous and hence difficult. However, the high uncertainty of CV selection favors an appropriate combination of classifiers! On the other hand, when one classifier is significantly better than others, combining it and poor classifiers can only damage the overall performance, and therefore selecting one classifier is expected to work better in this case than combining classifiers.

Jiang et al. (2005) combined bagging and graphical models. Their method starts from generating bootstrap samples from the training data. After that, it employes a causal discoverer to induce a dependency model of features, expressed as a directed acyclic graph for each sample. If a node has neither direct nor indirect connections to the class node in all the graphs, the corresponding feature is removed, i.e., it is considered as irrelevant to the class attribute on all the bootstrap samples. In this context, using graphical models is viewed as feature selection. Predictions of the individual classifiers trained from the samples are combined by majority voting. Though this approach can produce stronger ensembles than bagging alone, an evident weakness of this approach to building ensembles of classifiers is that the total computational cost is largely burdened by constructing a dependency model on each sample.

Though most of work on ensembles of NN classifiers concentrates on improving classification accuracy, Barandela et al. (2003) carried out experiments involving imbalanced data sets and outlier detection, and they obtained promising results, thus opening a new research line.

4. Our Approach

As one can see from the survey done in the previous section, all authors utilized the conventional NN classifier in building ensembles. In contrast to others, we chose a variant of the NN classifier called K-local hyperplane distance nearest neighbor (HKNN) (Vincent & Bengio, 2002), which demonstrated a competitive performance to the state-of-the-art (support vector machines) for handwritten digit (Vincent & Bengio, 2002) and protein fold recognition (Okun, 2004b). As the conventional NN classifier, HKNN does not require training. It computes distances of each test point x to L local hyperplanes, where L is the number of different classes. The ℓth hyperplane is composed of K nearest neighbors of x in the training set, belonging to the ℓth class. A test point x is associated with the class whose

hyperplane is closest to x. In general, K should not be too small, otherwise a fitting hyperplane may be not unique. Therefore, the approaches like (Alpaydin, 1997; Skalak, 1996) aiming at training set editing are inappropriate when many classes are sparsely represented in the training set. HKNN has two parameters, K and λ (penalty term), to be optimized.

It seems that feature selection is the most popular way to make individual classifiers diverse, because perturbing training patterns (bagging, boosting) does not work well for the NN classifiers. However, feature selection is typically time-consuming. Moreover, in certain cases, almost all features can be relevant for characterizing classes of patterns so that none of the features can be preferred to others. Instead of feature selection, we introduce multiple views, where each view is associated with its own individual classifier. Multiple views are automatically generated from feature sets describing patterns. Given that F is a set of the original feature sets, the total number of views formed from these sets, N_V, is $\sum_{i=1}^{|F|} \binom{|F|}{i}$. For example, if $|F| = 6$, then $N_V = 6 + 15 + 20 + 15 + 6 + 1 = 63$.

Our classification scheme is akin to stacked generalization (Wolpert, 1992), where a set of individual classifiers forms the first level while a single combining algorithm forms the second level. Though stacked generalization cannot always guarantee the improved classification performance, it can be used as a meta-learning technique to boost accuracy.

First, we need to select certain views from the complete set of views. Selection can be 1) random, 2) based on cross-validation errors, or 3) based on validation errors.

With the first option, one simply randomly chooses M views like in (Bay, 1998; Bay, 1999). However, random selection may not always guarantee the best performance, compared to a single classifier.

The second option assumes that the optimal parameters of individual classifiers are determined by cross-validation applied to the training data, and each view is linked to a certain CV error. Then one needs to pick M views, all associated with 1) small differences in CV errors, i.e., high instability in performance, and 2) sufficiently high accuracies, like done in (Paik & Yang, 2004). Compared to the first option, the second one reduces randomness of selection because of two constraints.

Finally, the third option assumes a separate validation set, which is used for a sensitivity analysis of views. In this case, one starts from the view resulting in the lowest CV error. Other views are iteratively added to this view, one at a time, based on the minimum validation error, i.e., the view led to the smallest error of an ensemble is linked up to the set of the previously selected views. Such a procedure is akin to the wrapper approach to feature selection with forward sequential selection. John et al. (1994) and Aha and Bankert (1994) demonstrated that the wrapper strategy is superior to the filter one, because it avoids the problem of using an evaluation function whose bias differs from the classifier. Thus, it seems that the third option is the best among the three, but quite often a separate validation set is not provided or it is impossible to collect it.

Based on this reason, we make use the second option, though in order to demonstrate the full potential of our approach, we will utilize the third option with the test set applied for validation (this will be merely done for illustration of our ideas).

The individual NN classifiers need to be combined together in order to make the final predictions. To combine outputs of these classifiers, which are simply class labels, a classification result vector (CRV) (Lepistö et al., 2003) is used. CRV is a nearest neighbor combining algorithm using the Hamming distance applied to the vectors of class predictions. CRV acts as a kind of ECOC. Unlike the latter, CRV does not need to convert the k-way classification problem into a set of binary problems. As follows from its name, ECOC is capable of correcting a certain number of errors, and this fact makes ECOC somewhat superior over majority voting. It is known that the NN classifiers do not benefit much from ECOC unless appropriate features are selected for each individual classifier (Ricci & Aha, 1998). Since multiple views constitute feature selection, CRV can be considered a good alternative to majority voting.

5. Data Set and Challenges It Presents

The real-world protein data set derived from the SCOP (Structural Classification of Proteins) database (Lo Conte et al., 2000) was used. This data set is available on line[1] and its detailed description can be found in (Ding & Dubchak, 2001). It contains the 27 most populated folds represented by seven or more proteins and corresponding to four major structural classes. The term "fold" is related to 3-D protein structure. Understanding of how proteins form such 3-D structures has tremendous impact in new drug design.

The training set consists of 313 protein folds having

[1] http://crd.lbl.gov/∼cding/protein

(for each two proteins) no more than 35% of the sequence identity for aligned subsequences longer than 80 residues. The test set of 385 folds is composed of protein sequences of less than 40% identity with each other and less than 35% identity with the proteins of the first dataset. In fact, 90% of the proteins of the second dataset have less than 25% sequence identity with the proteins of the first dataset. Such low identity both within each set and between two sets renders sequence analysis of proteins based on sequence-to-sequence comparisons completely useless. Hence, we turn to structure analysis based on physicochemical features extracted from protein sequences. Nevertheless, the significant difference between the training and test data together with multiple, sparsely populated classes presents the main challenge for achieving high classification accuracy.

Six feature sets in the data set are: amino acids composition (C), predicted secondary structure (S), hydrophobicity (H), normalized van der Waals volume (V), polarity (P), and polarizability (Z). All but the first feature set have dimensionality 21, whereas composition has dimensionality 20. In total, when combined together, six feature sets form 125-dimensional vectors for each fold. Since there are six feature sets, 63 views can be generated, where each view includes from one to six feature sets. For instance, CH means the view composed of composition and hydrophobicity feature sets while PSV stands for the view combining polarity, secondary structure, and volume feature sets.

6. Experiments

As mentioned before, Ding and Dubchak (2001) had already split the initial data into the training and test sets. As a result, the main efforts of researchers who used these sets were spent on classifying the test set, whereas the training set was primarily used for cross-validation in order to find the optimal parameters of the individual classifiers. We determined the optimal parameters (K and λ) of each HKNN classifier by means of leave-one-out cross-validation (loo-cv). Each feature was normalized to zero mean and unit variance (Okun, 2004a) prior to cross-validation.

First, we verified the hypothesis of Paik and Yang (2004) that classifiers with similar CV errors are good candidates for inclusion into an ensemble. For this we sorted views and respective classifiers according to loo-cv error and picked M ($3 \leq M \leq 9$) views/classifiers from different continuous segments of the sorted list, e.g., first or last nine views/classifiers. Our observation is that it was not always true than combining classifiers with similar CV errors leads to sizeable improvement in accuracy. When accuracy of the whole ensemble grew up, it was often less than 1%, which implies that not any ensemble was good. Thus, selection of proper views is of importance.

Table 1. Decrease in test error of the ensemble as the views were iteratively added by one at a time.

VIEW	ENSEMBLE TEST ERROR	INDIVIDUAL CLASSIFIER LOO-CV ERROR
CSV	44.67	48.56
CHPZS	42.60	55.27
CHPS	40.78	56.87
C	39.74	62.30
CPZV	38.70	61.66
CZV	37.14	65.50
CH	36.88	59.43
CS	36.62	53.04
HPS	36.36	66.13

Next, we checked the maximum accuracy gain when using the test set for validation[2]. That is, we started from the view (CSV in our case) leading to the lowest loo-cv error (48.56%) and added the next views iteratively, by one at a time, to CSV until the new minimum test error of the ensemble was smaller than the previous minimum error. After combining nine views, the test error rate fell to 36.36%. As for single-view classification, the lowest test error corresponds to 41.04% (CHPZS), which is 4.68% larger than the result achieved with multiple views. Our result is also smaller than the best result (38.80% in (Bologna & Appel, 2002)), involving ensembles of classifiers based on bagging neural networks.

Table 1 summarizes results. The first column contains the views added one after another starting from CSV. That is, CHPZS was added to CSV, followed by CHPS, etc. The second column lists test errors of the ensemble each time when a new view was added. The third column shows loo-cv errors related to the individual classifiers incorporated into the ensemble. As one can see, these loo-cv errors are quite diverse, and this fact did not prevent the whole ensemble to reach very low test error.

[2]Of course, it would be possible to partition the training set in order to have a proper validation set. However, the sparse training set of dimensionality, relatively high compared to the number of training patterns, prevents this option because of the curse of dimensionality. The same argument is applied to the test set. In addition, the HKNN performance would quickly degrade if the number of nearest neighbors, K, is too small.

7. Conclusion

Multi-view classification was considered in the context of ensembles of NN classifiers. Replacing feature selection with multiple views, it is possible to dramatically lower computational demands to combining classifiers. At the same time, as demonstrated by experiments with protein fold recognition, accuracy improvement can be achieved (up to 4.68% in our study), thus contributing to research on ensembles of NN classifiers.

Acknowledgments

Authors are thankful to anonymous reviewers for valuable comments that helped to significantly improve the paper.

References

Abney, S. (2002). Bootstrapping. *Proceedings of the Fortieth Annual Conference of the Association for Computational Linguistics, Philadelphia, PA* (pp. 360–367).

Aha, D., & Bankert, R. (1994). Feature selection for case-based classification of cloud types: an empirical comparison. *Proceedings of the AAAI Workshop on Case-Based Reasoning, Seattle, WA* (pp. 106–112).

Alkoot, F., & Kittler, J. (2002a). Moderating k-NN classifiers. *Pattern Analysis and Applications*, *5*, 326–332.

Alkoot, F., & Kittler, J. (2002b). Modified product fusion. *Pattern Recognition Letters*, *23*, 957–965.

Alpaydin, E. (1997). Voting over multiple condensed nearest neighbors. *Artificial Intelligence Review*, *11*, 115–132.

Bao, Y., & Ishii, N. (2002). Combining multiple k-nearest neighbor classifiers for text classification by reducts. In S. Lange, K. Satoh and C. Smith (Eds.), *Proceedings of the Fifth International Conference on Discovery Science, Lübeck, Germany*, vol. 2534 of *Lecture Notes in Computer Science*, 340–347. Springer-Verlag, Berlin.

Bao, Y., Ishii, N., & Du, X. (2004). Combining multiple k-nearest neighbor classifiers using different distance functions. In Z. Yang, R. Everson and H. Yin (Eds.), *Proceedings of the Fifth International Conference on Intelligent Data Engineering and Automated Learning, Exeter, UK*, vol. 3177 of *Lecture Notes in Computer Science*, 634–641. Springer-Verlag, Berlin.

Barandela, R., Sánchez, J., & Valdovinos, R. (2003). New applications of ensembles of classifiers. *Pattern Analysis and Applications*, *6*, 245–256.

Bay, S. (1998). Combining nearest neighbor classifiers through multiple feature subsets. In J. Shavlik (Ed.), *Proceedings of the Fifteenth International Conference on Machine Learning, Madison, WI*, 37–45. Morgan Kaufmann.

Bay, S. (1999). Nearest neighbor classification from multiple feature sets. *Intelligent Data Analysis*, *3*, 191–209.

Bickel, S., & Scheffer, T. (2004). Multi-view clustering. *Proceedings of the Forth IEEE International Conference on Data Mining, Brighton, UK* (pp. 19–26).

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, Madisson, WI* (pp. 92–100).

Bologna, G., & Appel, R. (2002). A comparison study on protein fold recognition. *Proceedings of the Ninth International Conference on Neural Information Processing, Singapore* (pp. 2492–2496).

Breiman, L. (1996a). Bagging predictors. *Machine Learning*, *24*, 123–140.

Breiman, L. (1996b). *Bias, variance, and arcing classifiers* (Technical Report 460). University of California, Statistics Department.

Ding, C., & Dubchak, I. (2001). Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics*, *17*, 349–358.

Freund, Y., & Schapire, R. (1996). Experiments with a new boosting algorithm. In L. Saitta (Ed.), *Proceedings of the Thirteenth International Conference on Machine Learning, Bary, Italy*, 148–156. Morgan Kaufmann.

Jiang, Y., Ling, J.-J., Li, G., Dai, H., & Zhou, Z.-H. (2005). Dependency bagging. In *Proceedings of the Tenth International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, Regina, Canada*, Lecture Notes in Artificial Intelligence. Springer-Verlag, Berlin.

John, G., Kohavi, R., & Pfleger, K. (1994). Irrelevant features and the subset selection problem. In W. Cohen and H. Hirsh (Eds.), *Proceedings of the Eleventh International Conference on Machine Learning, New Brunswick, NJ*, 121–129. Morgan Kaufmann.

Kailing, K., Kriegel, H.-P., Pryakhin, A., & Schubert, M. (2004). Clustering multi-represented objects with noise. In H. Dai, R. Srikant and C. Zhang (Eds.), *Proceedings of the Eighth Pasific-Asia Conference on Knowledge Discovery and Data Mining, Sydney, Australia*, vol. 3056 of *Lecture Notes in Artificial Intelligence*, 394–403. Springer-Verlag, Berlin.

Lepistö, L., Kunttu, I., Autio, J., & Visa, A. (2003). Classification of non-homogeneous texture images by combining classifiers. *Proceedings of the IEEE International Conference on Image Processing, Barcelona, Spain* (pp. 981–984).

Lo Conte, L., Ailey, B., Hubbard, T., Brenner, S., Murzin, A., & Chotia, C. (2000). SCOP: a structural classification of proteins database. *Nucleic Acids Research*, *28*, 257–259.

Okun, O. (2004a). Feature normalization and selection for protein fold recognition. *Proceedings of the Eleventh Finnish Artificial Intelligence Conference, Vantaa, Finland* (pp. 207–221).

Okun, O. (2004b). Protein fold recognition with k-local hyperplane distance nearest neighbor algorithm. *Proceedings of the Second European Workshop on Data Mining and Text Mining for Bioinformatics, Pisa, Italy* (pp. 47–53).

O'Sullivan, J., Langford, J., Caruana, R., & Blum, A. (2000). FeatureBoost: a metalearning algorithm that improves model robustness. In P. Langley (Ed.), *Proceedings of the Seventeenth International Conference on Machine Learning, Stanford, CA*, 703–710. Morgan Kaufmann.

Oza, N., & Tumer, K. (2001). Input decimation ensembles: decorrelation through dimensionality reduction. In J. Kittler and F. Roli (Eds.), *Proceedings of the Second International Workshop on Multiple Classifier Systems, Cambridge, UK*, vol. 2096 of *Lecture Notes in Computer Science*, 238–247. Springer-Verlag, Berlin.

Paik, M., & Yang, Y. (2004). Combining nearest neighbor classifiers versus cross-validation selection. *Statistical Applications in Genetics and Molecular Biology*, *3*, Article 12.

Ricci, F., & Aha, D. (1998). Error-correcting output codes for local learners. In C. Nedellec and C. Rouveirol (Eds.), *Proceedings of the Tenth European Conference on Machine Learning, Chemnitz, Germany*, vol. 1398 of *Lecture Notes in Computer Science*, 280–291. Springer-Verlag, Berlin.

Rüping, S. (2005). Classification with local models. In K. Morik, J.-F. Boulicaut and A. Siebes (Eds.), *Proceedings of the Dagstuhl Workshop on Detecting Local Patterns*, Lecture Notes in Computer Science. Springer-Verlag, Berlin.

Schapire, R., Freund, Y., Barlett, P., & Lee, W. (1997). Boosting the margin: a new explanation for the effectiveness of voting methods. In D. Fisher Jr. (Ed.), *Proceedings of the Fourteenth International Conference on Machine Learning, Nashville, TN*, 322–330. Morgan Kaufmann.

Skalak, D. (1996). *Prototype Selection for Composite Nearest Neighbor Classifiers*. PhD thesis, Department of Computer Science, University of Massachusetts.

Tsochantaridis, I., & Hofmann, T. (2002). Support vector machines for polycategorical classification. In T. Elomaa, H. Mannila and H. Toivonen (Eds.), *Proceedings of the Thirteenth European Conference on Machine Learning, Helsinki, Finland*, vol. 2430 of *Lecture Notes in Computer Science*, 456–467. Springer-Verlag, Berlin.

Tsymbal, A. (2002). *Dynamic Integration of Data Mining Methods in Knowledge Discovery Systems*. PhD thesis, University of Jyväskylä.

Vincent, P., & Bengio, Y. (2002). K-local hyperplane and convex distance nearest neighbor algorithms. In T. Dietterich, S. Becker and Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems 14*, 985–992. MIT Press, Cambridge, MA.

Wolpert, D. (1992). Stacked generalization. *Neural Networks*, *5*, 241–259.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. *Proceedings of the Thirty-Third Annual Conference of the Association for Computational Linguistics, Cambridge, MA* (pp. 189–196).

Zhou, Z.-H., & Yu, Y. (2005a). Adapt bagging to nearest neighbor classifiers. *Journal of Computer Science and Technology*, *20*, 48–54.

Zhou, Z.-H., & Yu, Y. (2005b). Ensembling local learners through multimodal perturbation. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, *35*, 725–735.

Using Unlabeled Texts for Named-Entity Recognition

Marc Rössler MARC.ROESSLER@UNI-DUISBURG.DE
Univ. Duisburg, Lotharstr.65, 47048 Duisburg, Germany

Katharina Morik MORIK@LS8.CS.UNI-DORTMUND.DE
Univ. Dortmund, Baroper Str.301, 44227 Dortmund, Germany

Abstract

Named Entity Recognition (NER) poses the problem of learning with multiple views in two ways. First, there are not enough labeled texts so that the exploitation of unlabeled texts becomes necessary. Second, words and word sequences offer several aspects for representation, each reflecting another aspect of them. Instead of choosing the most promising representation as done in feature selection, the cooperation of different features enhances learning NER. In this paper, we investigate the bootstrapping of features. From labeled and unlabeled texts, features are determined which in turn are exploited to recognize names automatically. The SVM is used as the learning engine. Results on German texts and on biomedical texts show that the approach is promising.

1. Introduction

A name or Named Entity (NE) is a noun phrase which has almost no meaning besides its reference to a specific entity. Its recognition within texts is important, for instance, if we want to extract from documents all information about a particular person, location, or organisation. Named Entity Recognition (NER) is eased by linguistic tools and word lists. However, these are tedious to develop and domain sensitive. Hence, applying machine learning approaches to NER has become a research subject.

Although one name often consists of several words, the learning task is formulated as a classification of single words. Every word is classified into the predefined name categories and the additional is-not-a-name class. As shown in Table 1, this notation assumes that all adjacent items tagged with the same class-label form part of one single name (here: "Foreign Ministry" is an organisation and "Shen Guofong" is a person). The evaluation of NER is based on an exact matching scheme, i.e. also partially correctly tagged names count as mismatch (e.g., tagging only "Ministry" as ORG would be counted as complete failure).

Test and training sets for learning are prepared from all candidate words for the given categories. An instance consists of the word, its features, and the correct category (label). Among the learning methods are Hidden Markov Models (Bikel et al., 1997), Maximum Entropy Models (Borthwick et al., 1998), Support Vector Machine (SVM) (Takeuchi & Collier, 2002), Boosting and Voted Perceptron (Collins & Singer, 1999).

Features for names can represent different aspects: mere surface form of a word, linguistic knowledge about the word (morphological, syntactic, semantic), and statistics of the occurrence of the word in a document collection. In a pure learning setting, linguistic knowledge and, hence, the linguistic features are not available. The underlying linguistic processes are to be reflected by the form of a word, its context, and the frequency of occurrences. These are three different aspects or views. According to McDonald (McDonald, 1996), there are two complementary kinds of evidence for NER: Internal evidence is taken from within the NE (e.g., beginning with an uppercase letter), while external evidence is provided by the context in which a name appears. We add as a third aspect the frequency of occurrences.

Frequency of words or contexts in labeled texts are the basis of learning NER. However, labeled corpora are hard to get for each language and domain pair. Therefore, a co-training scheme is promising (Blum & Mitchell, 1998). Its presupposition that the feature types are orthogonal is given. Hence, the frequencies within unlabeled texts can be used, as well. We may

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

Table 1. The word-tag assignment to NE

Foreign	Ministry	spokesman	Shen	Guofong	announced	in	Taipei
ORG	ORG	0	PER	PER	0	0	LOC

consider external and internal evidence two independent hypotheses on a name being of a particular NE category. Only if both agree, the word is classified accordingly. Following Steven Abney, this procedure can be applied to unlabeled data (Abney, 2002).

We may illustrate the NER learning procedure using different aspects by the following example. An approach focussing on internal evidence combines an uppercase beginning with a simple lookup in a list of known names. It classifies some names correctly, depending on the size of the lists. However, it ignores lexical ambiguity, e.g. Philip Morris as person or organisation. It cannot cover all names, especially names of the category ORGANISATION, because new companies are founded. External evidence, the context of the word to classify, helps to classify ambiguous or unknown words. For instance, "is headquartered in" often follows a company and precedes a location. To make estimations on the external evidence feasible, a sliding window approach can be used: Only a fixed span of words, for instance three words before and two words after the item to classify are considered. An example is given in Table 2. This technique is applied to all words that could be part of a NE. If external evidence is found for a sequence of words predicting the category of the succeeding word, this sequence is entered into a context lexicon. Using this, in turn, classifies names in the unlabeled corpus. This enlarges the list of known words. Now, more names can be classified. In this way, internal and external evidence can be used in a bootstrapping manner.

Due to the strict matching scheme used for the evaluation of NER systems, it is necessary to recognize the whole sequence of NEs. In order to focus on the detection of the correct boundaries, we have developed an enhanced sliding window approach. Keeping the context window of three preceding and two succeeding words fixed, we expand the instance to classify dynamically from one up to eight words. Although this enhanced sliding window approach augments the number of instances notably, the F-Measure is almost identical to the approach for classifying single words. However, precision is much higher while recall is lower. An example for the enhanced sliding windows is given in Table 3.

In this paper, we investigate the bootstrapping of features for NER learning. Our approach is evaluated by the tasks of NER for German and for the biomedical domain. Both, NER for German and for the biomedical domain is more challenging than the classical NER, because capitalization is not a reliable indicator for the detection of names. In German all nouns, and not just names are capitalized; within English texts of the biomedical domain, only some names start with an uppercase letter. This increases the number of name candidates and the lexical ambiguity. The free word order of the German language makes NER even more difficult, because the context is varying more.

2. Exploiting unlabeled data for bootstrapping features

Our approach refrains from any handcrafted lists and any linguistic processing. Given are an annotated corpus, which is divided into training and test set, and an unlabeled corpus of the same domain/language. According to the word frequencies, words are determined which are not considered candidates for part of a name. For all other words instances are formed for each category (PERSON, LOCATION, ORGANISATION). An instance contains the correct tag and the features of the word and those of its context. The features can be grouped into five sets

f1: Deterministic word-surface features like, e.g., "4-digit number", "Capitalized", "Uppercase only" etc.

f2: Character-based word length

f3: Sub-word-form representation with positional substrings. The word "Hammer" is represented as: "r", "er", "mer" at last position,"ham" at first, "amm" at second position etc.

f4: Corpus-lexicon representing how often a word was seen as NE of a particular category.

f5: Context-lexicon listing word sequences preceding and succeeding NEs.

The first three feature sets describe the surface of a single word. From the first set, the 20 surface features, only one can apply to a certain word. The second feature type is just the word length in characters.

Table 2. The sliding window approach

context considered			focus	context considered	
FOREIGN	MINISTRY	SPOKESMAN	SHEN	GUOFONG	ANNOUNCED
MINISTRY	SPOKESMAN	SHEN	GUOFONG	ANNOUNCED	IN

The third feature set splits the word into substrings together with their position. Up to 8 such positional substrings can apply to a word. The fourth and fifth feature set are based on an unlabeled corpus. They describe all occurrences of a word within all considered unlabeled texts. For the 6 words within the simple sliding window, 42 feature values can maximally be given. Hence, an instance is a vector of maximal length 252, but being sparse the longest vector in our applications had length 192.

The fourth feature set, f4, is automatically created using unlabeled data. Given classifiers for the categories trained on an annotated corpus, they can be used to create lexical resources by applying them to unlabeled text. The classifier trained on PERSON, for instance, returns for each word occurrence x_i a numerical value $f(x_i)$. If $f(x_i)$ is greater than 0, the word occurrence is labeled positively as PERSON. The output of the classifiers will be full of false negatives but will also contain a lot of true positives. Since these true positives are based on the few names and contexts learned from the labelled data, they can be used to extract new names. The function value $f(x_i)$ for each word occurrence in the unlabeled corpus is discretised into some intervals c_j. To reflect lexical ambiguity, for all words occurring in the unlabeled data, the assignments of the different c_j are counted and those with a frequency ratio

$$\frac{freq(c_j)}{freq(x)} \geq \theta$$

are turned into features. For instance, a word x with $freq(x) = n$ may receive by the PERSON classifier m_1 times a function value in the range $c_1 = [-0.5, 0]$ and m_2 times $f(x)$ in $c_2 = [0.5, 1]$. If the two fractions exceed the threshold, then two features are constructed, namely

$$PERSONc_1 = \frac{m_1}{n} \qquad PERSONc_2 = \frac{m_2}{n}$$

All classifiers contribute to forming these features. Because the classifiers are not perfect, the numbers counted are far from being correct. Nevertheless they show tendencies of words to co-occur with particular labels. The evidence learned becomes the new feature set f4, which is used by training classifiers on the labelled data, again. The resulting classifiers are applied iteratively to the unlabeled data in order to extract more evidence. This bootstrapping procedure completes the representation of internal evidence.

After applying the classifiers to the unlabeled data, the NEs found are marked up in the raw corpus. Based on this markup, the context lexicon, feature set f5, is created. All word sequences up to three words are extracted and represented with information on the name class they immediately preceded or succeeded. All word sequences seen more than once as context of a particular name class are stored in the context lexicon, together with the relative frequency of occuring as context of a particular class divided by the total frequency of the word sequence.

3. Experiments

We used the training data published for the shared task on NER for the biomedical domain (Kim et al., 2004) and those of the German NER at the CoNLL 2003 shared task (Tjong Kim Sang & De Meulder, 2003). The sub-word form representation reduces the feature space significantly. The biomedical corpus contains 22.000 word forms and 11.000 substrings. Within our German data, we counted 33.000 word forms and reduced them to 14.000 substrings, without any frequency threshold. Still, we need a powerful and efficient learning algorithm to deal with this large feature set. We choose the linear SVM_light (Joachims, 1998), a learning algorithm for binary classification, which is able to handle large numbers of parameters efficiently. As common in NER, we stored all names recognized within one text and marked-up other, previously untagged occurrences of these names.

In a first system, we only evaluated the use of unlabeled texts for the learning of internal evidence. For all experiments we set the simple sliding window to six words, i.e. three preceding, the current, and two succeeding words. For German, experiments were conducted on the 200.000 words annotated for CoNLL 2003 shared task on NER. For bootstrapping, 40 Millions of words from the FR-Corpus were used (Frankfurter-Rundschau, 2004). The results for person names were impressive (Roessler, 2004b). Adding one simple rule to detect coordinated names, an F-Measure of almost 0.9 was scored, which equals the performance of the two rule-based systems for Ger-

Table 3. The enhanced sliding window approach

context considered			focus	context considered	
EBENSO	SCHNELL	HAT	BEAR	STEARN	KONKURRENZ
EBENSO	SCHNELL	HAT	BEAR STEARNS	KONKURRENZ	DIE
EBENSO	SCHNELL	HAT	BEAR STEARNS KONKURRENZ	DIE	NEUE

man (Neumann & Piskorski, 2002; Volk & Clematide, 2001) and easily outperformed all contributions to the shared task. This was scored only after two iterations of the bootstrapping cycle. However, the performance on the organisation and the location names reached a plateau on a rather low level. Note, however, that we compare our results with those achieved by using hand-crafted rules and word lists. Not using unlabeled data does not result in any compatible NER here.

The experiments on the biomedical domain (Roessler, 2004a) were conducted with an almost identical setting. Here, we investigated the effects of bootstrapping features. As unlabeled data, documents were taken from Medline delivering 100 Mio. Words. Our system scored an overall F-Measure of 0.64, which is in the lower middle field compared to the other systems participating at the shared task JNLPBA-2004 (Kim et al., 2004). Similar to German organisation and location names, the bootstrapping showed almost no effect to the 5 categories, which are not person names. We are convinced, that person names are much easier to bootstrap because of the clear syntax of first names and surnames (Quasthoff & Biemann, 2002). The other categories vary much more in their appearance. Bootstrapping with a focus on internal evidence of single words is not able to capture the multi-word character of names. Therefore, we used the enhanced sliding window approach in order to focus on the multiword units and to bootstrap contexts at the same time. Bootstrapping internal and external evidence at the same time also approximated a plateau on a rather low level. However, after every bootstrapping cycle the model learned was more compact in terms of the number of support vectors. We assume that this generalisation is based on a shift from specific knowledge about words to more general knowledge about contexts, learned from the unlabeled data.

Based on this observeration, we developped the architecture shown in Figure 1. Starting with the enhanced sliding window approach, we iterated bootstrapping to learn external evidence until a plateau was reached. Using the learned resources, we switched to the basic sliding window approach to learn internal evidence.

Although our approach outperformed the best contributions of the competition only in the category PERSON, the results convinced us that learning from unlabeled data is competible with using hand-crafted lists. Some words (e.g., "Treuhand"), however, cannot be recognised without name lists.

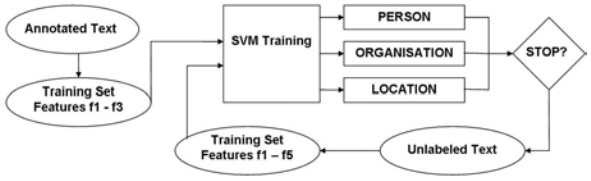


Figure 1. The process starts by creating instances for the enhanced sliding window approach using only the feature sets f1-f3. After the training of the SVMs, the classifiers are applied to the unlabeled data in order to create the feature sets f4 and f5 and the SVMs are retrained. After every iteration it is checked, whether there is still an improvement of the classifiers performance. The first time, when there is no further improvement, the simple sliding window replaces the enhanced sliding window approach. The second time, the algorithm stops.

4. Conclusion and Related work

We have shown a method which forms two lexicons for NER learning by applying learning results from labeled texts to unlabeled texts. For each name category, a classifier is learned. The classifiers are applied to the unlabled texts. The result is used to form

- the corpus lexicon which lists names together with their NE-categories, and
- the context lexicon which lists word sequences together with the categories they preceded or succeeded.

The obtained lexicons are used for further learning in a bootstrapping manner.

We used SVM-light as learning engine. SVM for biomedical NER were first applied by (Takeuchi & Collier, 2002), or recently by (Bickel et al., 2004) in combination with a context-sensitive post-processing to gather multi-word names. However, they did not

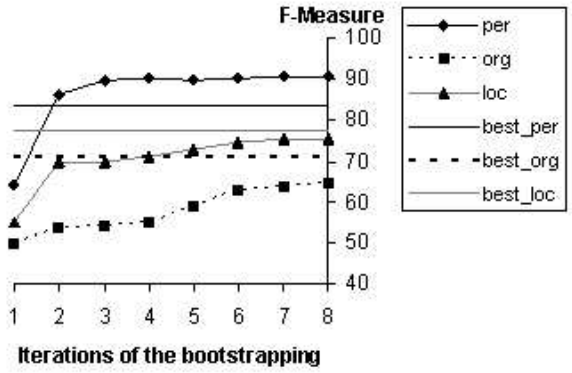


Figure 2. Comparing the bootstrapping of internal and external evidence with the best results scored at CoNLL. Iterations 1-4 are based on the enhanced , iterations 5-8 on the simple sliding window approach.

use unlabeled data. Varying the number of bootstrapping cycles we saw that for each name category a plateau is reached where further bootstrapping is no more effective. Observing the number of support vectors shows that bootstrapping makes the learned models more and more general or compact, that is, the number of support vectors is reduced from cycle to cycle.

Experiments have shown that this approach makes NER learning comparable with approaches that use linguistic tools or carefully craftes lexicons. What still remains unclear is the reason for the different impact of unlabeled data for PERSON and the other categories. Possible reasons are the free word order of German making the use of contexts more difficult, and the large variety of forms in the biomedical names making internal evidence hard to achieve.

Also other approaches to the classical NER task refraining from any lists of names (like (Mikheev A. & C., 1999; Zhou & Su, 2003)) showed results comparable to knowledge-rich systems. A very promising direction lies within the learning of names and triggers from unlabeled texts in a bootstrapping cycle. So-called seed lists, containing a few names, or seed rules, that reliably predict names, are used to learn internal and external evidence (Collins & Singer, 1999) (Riloff & Jones, 1999; Thelen & Riloff, 2002), (Lin et al., 2003), (Cucerzan & Yarowky, 2002; Cucerzan & Yarowky, 1999), (Ghani & Jones, 2002), (Quasthoff & Biemann, 2002). In contrary to these approaches we use an annotated corpus instead of seed lists or seed rules. Additionally, we are not working with different classifiers for different views but simply add and update features for one learner per category. Moreover, we always use the same sample of unlabeled data and do not select instances based on the classifers decisions. Like (Ghani & Jones, 2002) and (Quasthoff & Biemann, 2002) we deal with entities without a distinctive uppercase beginning. (Quasthoff & Biemann, 2002) report an experiment with EM-style bootstrapping of German person names. (Ghani & Jones, 2002) compare different bootstrapping algorithms for the semantic tagging of noun phrases.

References

Abney, S. (2002). Bootstrapping. *40th Annual Meeting of the Association for Computational Linguistics: Proceedings of the Conference*.

Bickel, S., Brefeld, U., Faulstich, L., Hakenberg, J.and Leser, U., Plake, C., & Scheffer, T. (2004). A support vector machine classifier for gene name recognition. In *BioCreative: EMBO workshop - a critical assessment of text mining methods in molecular biology*. Granada, Spain.

Bikel, D. M., Miller, S., Schwartz, R., & Weischedel, R. (1997). Nymble: a high-performance learning namefinder. *Proceedings of ANLP-97* (pp. 194–201).

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Annual Conference on Computational Learning Theory (COLT-98)*.

Borthwick, A., Sterling, J., Agichtein, E., & R., G. (1998). NYU: Description of the MENE named entity system as used in MUC-7. *Proceedings of the Seventh Message Understanding Conference*. Morgan Kaufmann.

Collins, M., & Singer, Y. (1999). Unsupervised models for named entity classification. *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Cucerzan, S., & Yarowky, D. (1999). Language independent named entity recognition combining morphological and contextual evidence. *Proceedings of the 1999 Joint SIGDAT Conference on EMNLP and VLC*. (pp. 132–138). College Park.

Cucerzan, S., & Yarowky, D. (2002). Language independent NER using a unified model of internal and contextual evidence. *Proceedings of CoNll-2002, The Sixth Workshop on Computational Language Learning*. Taipei, Taiwan, San Francisco: Morgan Kaufmann.

Frankfurter-Rundschau (2004). Corpus 1994. Published on the ECI Multilingual Text CD.

Ghani, R., & Jones, R. (2002). A comparison of efficacy and assumptions of bootstrapping algorithms for training information extraction systems. *Workshop on Linguistic Knowledge Acquisition and Representation: Bootstrapping Annotated Data, Resources and Evaluation Conference (LREC 2002).*

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Proceedings of the European Conference on Machine Learning* (pp. 137 – 142). Berlin: Springer.

Kim, J.-D., Otha, T., Yoshimasa, T., Yuka, T., & Collier, N. (2004). Introduction to the bio-entity recognition task at JNLPBA. *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications(JNLPBA-2004).* Geneva, Switzerland.

Lin, W., Yangarber, R., & Grishman, R. (2003). Bootstrapped learning of semantic classes from positive and negative examples. *Proceedings of the ICML 2003 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining.* Washington D.C.

McDonald, D. (1996). Internal and external evidence in the identification and semantic categorization of proper names. In B. Boguraev and J. Pustejovsky (Eds.), *Corpus processing for lexical acquisition*, 21–39. Cambridge, MA: MIT Press.

Mikheev A., M. M., & C., G. (1999). Named entity recognition without gazetters. *EACL'99* (pp. 1–8). Bergen, Norway.

Neumann, G., & Piskorski, J. (2002). A shallow text processing core engine. *Journal of Computational Intelligence*, *18*, 451–476.

Quasthoff, U., & Biemann, C. (2002). Named entitiy learning and verification: EM in large corpora. *CoNLL-2002, The Sixth Workshop on Computational Language Learning* (pp. 8–14). Taipei, Taiwan, San Francisco: Morgan Kaufmann.

Riloff, E., & Jones, R. (1999). Learning dictionaries for information extraction by multi-level bootstrapping. *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99).* Orlando, Florida.

Roessler, M. (2004a). Adapting an NER-system for German to the biomedical domain. *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004).* Geneva, Switzerland.

Roessler, M. (2004b). Corpus-based learning of lexical resources for German named entity recognition. *Proceedings of LREC 2004.* Lisboa, Portugal.

Takeuchi, K., & Collier, N. (2002). Use of support vector machines in extended named entity recognition. *Proceedings of CoNLL-2002, The Sixth Workshop on Computational Language Learning.* Taipei, Taiwan, San Francisco: Morgan Kaufmann.

Thelen, M., & Riloff, E. (2002). A bootstrapping method for learning semantic lexicons using extraction pattern contexts. *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing.* Philadelphia.

Tjong Kim Sang, E., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task - language independent named entitiy recognition. *Proceedings of CoNLL-2003.* Edmonton, Canada: Morgan Kaufmann.

Volk, M., & Clematide, S. (2001). Learn-filter-apply-forget - mixed approaches to named entity recognition. *Proceedings of the 6th International Workshop on Applicaions of Natural Language for Information Systems..* Madrid.

Zhou, G., & Su, J. (2003). Integrating various features in hidden markov model using constraint relaxation algorithm for recognition of named entities without gazetteers. *Proceedings of IEEE International Conference on NLP and KE* (pp. 732–739).

Interpreting Classifiers by Multiple Views

Stefan Rüping STEFAN.RUEPING@UNI-DORTMUND.DE
Universität Dortmund, LS Informatik 8, 44221 Dortmund, Germany

Abstract

Next to prediction accuracy, interpretability is one of the fundamental performance criteria for machine learning. While high accuracy learners have intensively been explored, interpretability still poses a difficult problem. To combine accuracy and interpretability, this paper introduces an framework which combines an approximative model with a severely restricted number of features with a more complex high-accuracy model, where the latter model is used only locally. Three approaches to this learning problem, based on classification, clustering, and the conditional information bottleneck method are compared.

1. Introduction

More and more data is collected in all kinds of application domains and sizes of data sets available for knowledge discovery increase steadily. On the one hand this is good, because learning with high-dimensional data and complex dependencies needs a large number of examples to obtain accurate results. On the other hand, there are several learning problems which cannot be thoroughly solved by simply applying a standard learning algorithm. While the accuracy of the learner typically increases with example size, other criteria are negatively affected by too much examples, for example interpretability, speed in learning and application of a model, overhead for handling large amounts of data and the ability to interactively let the user work with the learning system (Giraud-Carrier, 1998). This paper deals with the criterion of interpretability of the learned model, which is an important, yet often overlooked aspect for applying machine learning algorithms to real-world tasks. The importance of interpretability stems from the fact that knowledge discovery is not equal to the application of a learning algorithm, but is an iterative and interactive process that requires much manual work from data miners and domain specialists to understand the problem, transform the data prior to learning and interpret and deploy the model afterwards. One cannot hope to successfully solve these problems without substantial insight into the workings and results of the learning algorithm.

The rest of the paper is organized as follows: the next section discusses the concept of interpretability, its relation to multiple views, and introduces the basic ideas of this paper. Section 3 gives an introduction to the information bottleneck method, which will be used later. Section 4 describes the problem of learning local models and its connection to learning with multiple views, while three approaches to the crucial step of detecting local patterns are presented in Section 5. Following that, Section 6 gives some empirical results and Section 7 concludes.

2. Interpretability

The key problem with interpretability is that humans are very limited in the level of complexity they can intuitively understand. Psychological research has established the fact that humans can simultaneously deal with only about seven cognitive entities (Miller, 1956) and are seriously limited in estimating the degree of relatedness of more than two variables (Jennings et al., 1982). An optimal solution of a high-dimensional, large-scale learning task, however, may lead to a very large level of complexity in the optimal solution. Interpretability is very hard to formalize, as it is a subjective concept. In this paper, we use four heuristics to approach the concept of interpretability:

Number of features: the number of features used in a model is a heuristic measure of complexity. While this is not strictly true, as the user may understand even a high number of features if he can relate the feature values to an existing mental concept (e. g., a doctor may explain a large number of symptoms by one disease), this heuristic has

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

often been used in practice (Sommer, 1996).

User-defined hypothesis space: A very simple and yet very important finding in practice is that people tend to find those things understandable that they already know. Hence, if a user has much experience with a specific learning algorithm, it may be favorable to keep using this learner, regardless of its accuracy.

Examples and features instead of models: While models are often hard to understand even for experienced machine learning experts, single examples and single features have a clear meaning to domain experts (e. g. instead of a TF/IDF representation of text documents, one can look at the texts themselves).

Split-up into sub-problems: Splitting up a problem into several independent sub-problems reduces the complexity and may still give reasonable results, even if the sub-problems are actually not completely independent.

How can the interpretability problem be solved? Experience shows that one can often find a simple model which provides not an optimal solution, but a reasonably good approximation. The hard work usually lies in improving an already good model. Hence, we can try to find a simple model first and then concentrate on finding more sophisticated models only on those parts of the input space, where the model is not good enough. This will be an easier task because less examples have to be considered and hence one might use a more sophisticated learner. To express the fact that the latter models are only used for small parts of the input space, these models will be called local models. In contrast, the former, more general model will be called the global model (Rüping, 2005).

Implicitly, this setup requires a description of the parts of the input space where the global model is not good enough or a decision rule when to use the global model or the local models. These regions will be called local patterns (Hand, 2002) and we will require the description of the local patterns to be interpretable in the same sense as the global model. The idea, as depicted in Figure 1, is the following: to classify an observation with high accuracy, we see whether it falls into one of the local patterns. In this case, the corresponding local model is used, else the global model is used. When we are more interested in interpretability, it suffices to inspect only the global model, which is an approximation of the complete model, plus the local pattern which characterizes the deviations between the global and the complete model.

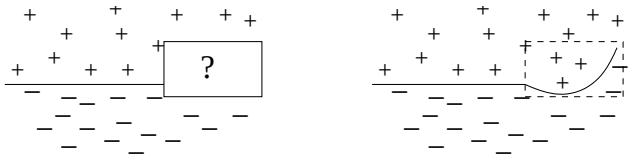


Figure 1. The local model idea. Left: with respect to interpretability, only the global model (horizontal line) is regarded, while the local pattern (rectangle) specifies the region where the global model is not reliable. Right: with respect to accuracy, the pattern specifies when to use the local model (nonlinear function) instead of the global model.

The local patterns distinguish this approach from both ensemble methods and from independently learning an understandable and a high-performance model. In usual ensemble methods, even in the case of easily interpretable base learners the combination of several learners will add a large amount of complexity, whereas using local patterns the model will be kept simple for a large and well-defined part of the input space. In contrast to two independently learned models, the local pattern assures that there is a strict, well-defined correspondence between the two models. The philosophy behind this approach is that accuracy and interpretability are two aspects of the same problem and that their solutions should be as independent as necessary, but as close as possible.

This approach reduces complexity in two ways. First, a less than optimal hypothesis language can be used for the global model, because errors can still be corrected by the local models. This leaves room for choosing a hypothesis language that optimizes criteria other than the prediction error, namely the interpretability of the global model. Second, for the aspect of discovering new knowledge, it may happen that the global model finds only the obvious patterns in the data that domain experts are already aware of. Patterns are more informative, if they contradict what is already known (Guyon et al., 1996). Hence, it may in fact be the case that the local models contain the interesting cases.

Of course, there is also a downside to this approach: The complete model, consisting of three sub-models, may easily be much more complex than a single model designed to optimize accuracy. However, it should be noticed that only the global model and the local patterns are meant to be understandable and that a lower complexity of a competing model yields no improvement if it is still too complex for the user to understand.

In this paper, the interpretability framework consists of restricting the number of features that both the learner and the description of the error regions of the learner may use. This can be seen as constructing two new views on the data: one to define the global model on, and one to discriminate between the global and the local models.

3. Information Bottleneck

The information bottleneck method (Tishby et al., 1999) extracts structure from the data by viewing structure extraction as data compression while conserving relevant information. With the data modeled by a random variable U[1], relevant information is explicitely modeled by a second random variable V, such that there is no need to implicitly model the relevant structure in terms of appropriately choosing distance or similarity measures as in standard clustering algorithm. The idea is to construct a probabilistic clustering, given by a random variable C, such that the mutual information $I(U, C)$ between the data and the clusters is minimized, i. e. C compresses the data as much as possible, while at the same time the mutual information $I(V, C)$ of the relevant variable V and the clusters is maximized, i. e. the relevant structure is conserved. Hence, the random variable C acts as a bottleneck for the information U has about V. Both goals are balanced against each other by a real parameter $\beta > 0$, such that the goal becomes to find a clustering $P(c|u)$ which minimizes

$$F = I(U, C) - \beta I(V, C).$$

It can be shown that this problem can be solved by iterating between the following three equations

$$\begin{aligned} P(c) &= \sum_u P(u)P(c|u) \\ P(v|c) &= \sum_u P(v|u)P(u|c) \\ P(c|u) &\propto P(c)e^{\beta P(v|u)\log P(v|c)}. \end{aligned}$$

The first two equations ensure the consistency of the estimate probabilities, while the third equation gives a functional form of the clustering, depending on the Kullback-Leibler-distance between $P(v|u)$ and $P(v|c)$ (removing factors independent of c). The input consists of the probability distributions $P(u)$ and $P(v|u)$.

[1]We use the letters U, V, W instead of the usual X, Y, Z in order to avoid confusion with the classification features X and labels Y used later

3.1. Condition Information Bottleneck

Gondek and Hoffmann (Gondek & Hofmann, 2004) extend the information bottleneck approach by considering not only information about relevant, but also about irrelevant structure. It is often easier to express what is already known and hence is uninteresting, than to specify what is interesting and relevant. Examples of such irrelevant structures are general categorization schemes and well-known properties of the data, when instead one is interested in how the data differs from what one thinks it looks like.

Conditional information bottleneck (CIB) is formulated by introducing a random variable W to describe the irrelevant information. The learning problem corresponds to that of standard information bottleneck with the new target function

$$F = I(U, C) - \beta I(V, C|W)$$

That is, one wants to maximize the information that C has of V, given that W is already known. In a way, the goal is to extract information orthogonal to what one can already infer via W.

Again, the problem can be solved by iterating between three estimation equations

$$\begin{aligned} P(c) &= \sum_u P(u)P(c|u) \\ P(v|w, c) &= \sum_u P(v|u, w)P(u|w, c) \\ P(c|u) &\propto P(c)e^{\beta \sum_w P(w|u) \sum_y P(v|u,w) \log P(v|w,c)} \end{aligned}$$

The probabilities $P(v|u, w)$, $P(w|u)$ and $P(u)$ have to be given to the learner as input.

4. Local Models and Multiple Views

Local pattern or subgroup detection (Hand, 2002) is defined as the un-supervised detection of high-density regions in the data. That is, one looks for regions in the input space, whose empirical probability with respect to a given a training set is significantly higher than the probability assigned by a default probability measure, which encodes the prior beliefs about the data. The idea is that the user already has some idea of what his data looks like and is interested in cases where his beliefs are wrong.

Local models are an extension of local patterns to the supervised case. Local models are meant to improve the prediction, hence instead of $P(x)$ the interesting quantity is the conditional class probability $P(y|x)$.

We will deal with classification rules only here. Given a global classifier $f(x)$, the goal is to find regions of the input space where $f(x)$ is wrong, plus a new, better classification rule on these regions. To justify the term *local* the error regions are restricted to have a probability below a user-defined threshold τ, such that most of the data will be predicted by the global model.

In order to improve interpretability with local models, we want both the global classifier and the description of the local regions to be interpretable. As discussed in Section 2, this implies that the user may choose any learner that he deems to be appropriate. Treating the learner as a black box, we improve understandability only by restricting the number of features for the global classifier and the local pattern detection can use. In other words, we construct a specific view for the global classifier which is optimized for interpretability of the overall model and a second view for the local patterns, which is optimized for describing the incorrectly predicted examples of the global model. Finally, the local classifier is not restricted in which features to use, as it is not meant to be understandable, but only to increase accuracy.

4.1. Optimizing the Global and Local Models

Selecting a subset of features for the global model is a well investigated problem and can be solved in a general way for example using the wrapper approach (Kohavi & John, 1998). This approach is computer intensive, but can be used for any learner.

The local learner is not restricted by any interpretability considerations at all and we may select the learner which gives the best accuracy. The definition of the local models asks only for the local model to be defined on its corresponding local pattern. Hence, we may use different learners for each pattern, which may be a good idea when there are different reasons that the data deviates from the global model. This means that the detection of local patterns and the construction of models depend on each other and that it may be a good idea to construct them in parallel or let them iteratively improve each other (Rüping, 2005).

However, as in the following we are mainly concerned with the problem of finding adequate local patterns, we simplify things by using only one local model on all local patterns. This model will be learned on all available examples (that is, it is actually a global model), but will be used only on the detected local patterns. This model is expected to perform better than the actual global model because it is not restricted by interpretability constraints.

5. Detecting Local Patterns

Given the global and local models, the goal of local pattern detection in this case is to find a description of the regions in the input space where the local model is better than the global one. Examples lying in these regions will be called local examples here. The goal of local pattern detection here is to optimize the combined learners accuracy while keeping the restriction of interpretability (hypothesis language and number of features) and locality (only a fraction of τ examples in the local patterns).

5.1. Local Patterns as a Classification Task

It is straightforward to define local pattern detection as a classification task: given the examples $(x_i, y_i)_{i=1}^n$ and the global and local learners predictions, define the new label l_i as 1 when (x_i, y_i) is a local example (meaning that the global learner predicts (x_i, y_i) wrong and the local learner is right). Set $l_i = -1$ otherwise. Then learn a classifier using the new labels. This classifier will predict whether the local model should be used on an example instead of the global one.

When the global and local learner agree, it obviously does not matter which prediction one uses. However, this does not mean that these examples can be removed from the local pattern learners training set. As the locality restriction requires that only a fraction of τ examples may lie in the local patterns, it is advisable to include only the local examples in the positive class, where the combined model can be improved by the local model. If the locality restriction is still not met, one will have to reduce the number positive predictions of the local pattern learner even more, e.g. by selecting a proper threshold for learners with a numerical decision function.

For the decision, which classifier to use for the local pattern task, the same interpretability considerations as for the global model apply. In fact, as may be advisable to use the same learner for both tasks. Letting the learner choose different sets of features and a different hypothesis in both tasks may provide enough variation to significantly improve the results.

5.2. Clustering Local Examples

Although in the strict sense detecting local patterns is a classification task in the framework of local models, it can also be solved by a clustering approach. The reason is, that the classification task is actually a quite relaxed one: given that the local model is more accurate than the global one and as long as the locality restriction is fulfilled, it is no problem to include more

examples in the local pattern than necessary. Hence, in this case the performance criterion for the classification is biased very much towards completely covering the positive class (recall) with less emphasis on the negative class.

This task may be solved by clustering the local examples using a density-based clusterer and choosing a threshold on the density to define the regions with highest probability of finding a local example. It is straightforward to optimize these thresholds such that the accuracy of the combined model is maximized.

A clustering approach may be better suited as a classification approach, as clustering not only tries to describe the differences between the local and the non-local examples, but actually tries to find a compact representation of the local examples. This description may give the user a clue why these examples are more complex to classify and may lead to an improved representation of the data.

One can also account for the probabilistic nature of the division of the examples into local and non-local examples. Assume that the training data $(x_i, y_i)_{i=1\ldots n}$ is i. i. d. sampled from $P_{orig}(X \times Y)$[2]. We start by learning a probabilistic classifier f, that is, a classifier whose outputs can be interpreted as the conditional class probability $f(x) = P_{orig}(Y = 1|x)$. Many classifiers either directly give such a probability or give outputs which can be appropriately scaled (Garczarek, 2002; Platt, 1999). We assume that this is the true distribution of positive and negative classes given the observation x and thus arrive at an estimate of $P_{orig}(Y \neq f(x)|x)$. Assuming $P_{orig}(x) = 1/n$ for all x in the training set gives an estimate of

$$P(x) := P_{orig}(x|Y \neq f(x)) = \frac{P_{orig}(Y \neq f(x)|x)}{\sum_x P_{orig}(Y \neq f(x)|x)}$$

the probability of drawing an falsely classified observation. When generating a cluster model, this probability can be used as a weight on how much each example will have to be represented by the cluster.

To enforce the restriction on the number of features used for the model, one can either use a clustering algorithm that incorporates feature reduction by selecting a subset of features that maximizes the density of the projection in this subspace (projected clustering, (Aggarwal et al., 1999)), or by selecting features after the clustering process, for example based on the mutual information $I(x, c)$ of the features x and the cluster membership values c.

[2]In the following, the index *orig* is also used to identify the marginal distributions derived from P_{orig}

5.3. Informed Clustering for Local Patterns

Informed clustering describes the setting where the desired structure to be extracted by a clustering is not only defined implicitly using the distance or similarity function, but also explicit information about relevant and irrelevant structure is given. The conditional information bottleneck algorithm is one such approach, where one can explicitely define irrelevant structure which the clustering algorithm should ignore.

There are two kinds of irrelevant information one could exploit. First, one can use a probabilistic clustering $p(c|x)$ of the complete training observations $(x_i)_{i=1\ldots n}$. This clustering shows, what the data generally looks like and can be used as a background model to discriminate the local examples against. Note that we do not require an information bottleneck clustering at this stage, we could also use any other probabilistic clusterer. It would also be possible to use an existing description of the data set at this stage (e. g. an ontology given by the user).

The other method is to define the prediction of the global model or, more precisely the conditional class probability $P_{orig}(Y = 1|x)$ as irrelevant. The idea here is that the global model is used anyway and that it is better to look for independent sources of information.

In either case, one can arrives at a well-defined probability $P_{cib}(w|u)$. Now one can set up the conditional information bottleneck problem to find the local patterns as follows: identify U with the index i and the relevant features V with the classification features X:

- $P_{cib}(u) = P(x_i) = P_{orig}(x_i|Y \neq f(x_i))$
- $\forall w : P_{cib}(v|u, w) = P_{ib}(v|u)$

In other words, the problem is to compute a probabilistic clustering $P(c|u) = P(c|x_i)$ of the observations x_i which are misclassified by the classifier f (controlled by $P_{cib}(u) = P_{orig}(x_i|Y \neq f(x_i))$), such that the clustering describes how the local examples differ from the complete data set (via defining the cluster information $P_{ib}(v|u)$ of the complete data set as irrelevant) or how the local examples differ from structure from the global model (via $P_{orig}(Y = 1|x)$).

As the information bottleneck method does not return a cluster model, but only the cluster membership probabilities $p(c|x)$, a model that induces these memberships has to be found in order to apply the clustering to new observations. Following the goal of interpretability, it is advantageous to use a k-medoids clustering model, as it is often easier to interpret single examples than models. For each cluster, we choose

that example as medoid, which – in the space of projections on the most relevant features – minimizes the expected distance of the medoid to the examples in the cluster, where expectation is taken with respect to the probability $P_{cib}(x, c)$ for the cluster c.

6. Experiments

In this section, we report results for both an instructive application for classifying music data, which we report in depth, and for a set of standard data sets in order to compare the different local pattern approaches.

6.1. Music Data

In these experiments, a linear Support Vector Machine (Vapnik, 1998) was used as both global and local classifier. Feature selection for the global classifier was performed by repeatedly removing the features with lowest absolute weight in the decision function. The SVM decision functions were probabilistically scaled using the method of (Platt, 1999). The pattern detection based on the CIB method (see Section 5.3) was used, where the initial clustering was obtained by standard information bottleneck and a k-medoids model of the CIB membership values was generated with the cosine similarity measure.

The data set in this experiment consists of 1885 audio files of songs from 8 music genres, combined with user reviews of each of the song as plain text. The classification target was to predict the music taste of a user. From the music files, 50 audio features were generated following the methodology of (Mierswa & Morik, 2005). The text information was represented as frequencies of the 500 most frequent words. The global classifier was learned from the text data only, as it is easy for users to interpret single keywords, while audio features are hard to understand even for experienced experts. The local classifier was learned on the union of the audio and text features. Notice that in this application we have four different views on the data: the keywords from the classification, the keywords from the clustering, the union of the audio and text features for the local classifier and finally the actual songs from the audio files, which the user can actually listen to.

The initial information bottleneck clustering was parameterized to return 8 cluster, in correspondence with the 8 music genres, and its parameter β was set to maximize the correspondence to the genres. However, the most informative words regarding the clustering were HELLO, POWER, BLEND, SOUNDS, BABY, FAT, QUIET, BIT, NIGHT, and GIVE, which do not seem to reveal any obvious genre structure.

Feature selection for classification returned GROOVE, SMOOTH, CHILL, JAZZY, MOOD, FUSION, PIANO, PIECE, PAUL, and JAZZ as the most important features. It is obvious that a certain music taste can be associated with these keywords

The CIB clustering returned TALENT, BABY, SOUNDS, CHECK, NEAT, PASS, TRUE, NICE, SEXY, and CHORUS as the most important features. Interestingly, extracting two medoids from this clustering showed that the first medoid consists only of the word CHORUS with no occurrence of the other keywords and the second medoid consists of the words SOUNDS and NICE with no occurrence of the other keywords. This is a result of the sparse structure of the text data, as a medoid as any other example will have only a few nonzero features. For sparse data it may be instructive to try out a different procedure to describe the CIB clusters. However, the second medoid with the keywords "sounds nice" seems to indicate that there are two aspects to musical taste in this data set, the genre – which the initial clustering was optimized against – (the classifier indicates that the user seems to like jazz) – and the quality of the music independent of the style (whether the song sounds nice or not).

5-fold cross-validation showed an accuracy of the global model of 0.624 ($\sigma = 0.0284$), while the local model achieved an accuracy of 0.670 ($\sigma = 0.0164$) measured over all examples. The combined model achieves an accuracy of 0.649 ($\sigma = 0.0230$). This lies between the accuracies of the global and the local model, which was expected, as the amount of examples that the global and the combined differ on is bounded by the parameter τ (in this experiment, $\tau = 0.05$), which stops the local model from correcting more errors of the global model.

To validate that the increase in performance is indeed a result of the conditional information bottleneck approach, the experiment was repeated with a standard information bottleneck clustering of the global models errors instead of the CIB step (all other parameters left constant). With the same accuracies for the global and local classifiers, the accuracy of the combined classifier dropped to 0.627 ($\sigma = 0.0329$). This proves that the conditional information bottleneck clustering finds novel structure in the errors of the global classifier.

To validate the effect of the parameter τ and the number of features for the CIB clustering, more experiments were conducted. The result can be seen in Table 1. The table shows the accuracies of the global, local and combined models and the disagreement rate (fraction of examples classified differently) between the global and the combined model. We can see that the

Table 1. Influence of the parameters on the performance.

PARAMETERS		ACCURACY			DISAGREE
τ	#FEATURES	GLOBAL	LOCAL	COMBINED	
0.05	10	0.624	0.670	0.648	0.147
0.05	20	0.624	0.670	0.653	0.201
0.025	10	0.646	0.670	0.642	0.070
0.025	20	0.646	0.670	0.646	0.019

combined model performs better when more features for the CIB clustering are present. We also see that the actual disagreement rate is higher than the given threshold τ. This is again a result of the sparse nature of the data, as in the space projected on the most important keywords, several different examples fall together, which prevents a more fine grained control of the number of local examples. An obvious tradeoff between interpretability in terms of number of features and accuracy can be observed here.

6.2. Standard Data Sets

To compare the local pattern approaches, the classification approach using a linear SVM, the clustering approach using an information bottleneck clusterer, and the informed clustering approach using conditional information bottleneck with the global classifiers conditional class probability estimate as irrelevant information were compared on a total of 8 data sets. 6 of the data sets (diabetes, digits, liver, balance, wine and breast-cancer) were taken from the UCI repository of machine learning databases (Murphy & Aha, 1994), and 2 additional real world data sets involving business cycle prediction (business) and intensive care patient monitoring (medicine) were used. The following table sums up the description of the data sets:

Name	Size	Dimension
balance	576	4
breast	683	9
diabetes	768	8
digits	776	64
liver	345	6
wine	178	13
business	157	13
medicine	6610	18

In these experiments, a linear Support Vector Machine (Vapnik, 1998) was used as both global and local classifier. Feature selection for the global classifier was performed by repeatedly removing the features with lowest absolute weight from the decision function until only 10% of the features were left.

Table 2 shows the experiments results, namely the accuracy of the global model, the local model (viewed as a complex global classifier and evaluated on the complete data set) and the combined models using classification, clustering and informed clustering for the local patterns. All results were 5-fold cross-validated. The performance of the combined model lies between the performances of the global and local models, which shows that local models can improve classification performance even under severe restrictions on the number of features (depending on the dimension of the data set, in some case only 1 feature is used). It can, however, not beat a more complex classifier in this case, which is a result of both the locality restriction stopping the local model from classifying more observations and the dimensionality restriction allowing only very coarse local patterns.

Overall, the clustering approach seems to be slightly better than the other, but differences are very small. This may be a sign that the local pattern detection task is essentially limited by the allowed size and complexity of the patterns in terms of number of features and not by the algorithm.

7. Conclusions

Next to accuracy, interpretability is a primary quality criterion for classification rules. Traditionally, these goals have been pursued independent of another. This paper showed that using the framework of local models, it is possible to find a model which combines not only good performance with an easily interpretable approximation, but, even more important, allows to give guarantees about the correspondence of the approximation and the combined classifier in terms of the disagreement rate threshold τ and in terms of an interpretable description of the error regions.

In the proposed algorithm, clustering proves to be an efficient tool for not only discriminating between the global and the local model, but also for describing the difference of both models in terms of a compact representation of the structure in the global models errors.

Table 2. Experimental Results.

Name	Global	Local	combined		
			SVM	IB	CIB
balance	0.644	0.940	0.727	0.909	0.788
breast	0.907	0.969	0.931	0.956	0.951
diabetes	0.760	0.781	0.701	0.748	0.763
digits	0.993	0.996	0.994	0.993	0.994
liver	0.579	0.695	0.635	0.634	0.631
wine	0.927	0.971	0.927	0.943	0.932
business	0.828	0.866	0.828	0.834	0.821
medicine	0.719	0.744	0.749	0.748	0.756

References

Aggarwal, C. C., Procopiuc, C., Wolf, J. L., Yu, P. S., & Park, J. S. (1999). Fast algorithms for projected clustering. *Proceedings of the ACM SIGMOD Conference on Management of Data* (pp. 61–72).

Garczarek, U. (2002). *Classification rules in standardized partition spaces*. Doctoral dissertation, Universität Dortmund.

Giraud-Carrier, C. (1998). Beyond predictive accuracy: What? *ECML'98 Workshop Notes - Upgrading Learning to the Meta-Level: Model Selection and Data Transformation* (pp. 78–85). Technical University of Chemnitz.

Gondek, D., & Hofmann, T. (2003). Conditional information bottleneck clustering. *Proceedings of the 3rd IEEE International Conference on Data Mining, Workshop on Clustering Large Data Sets*.

Gondek, D., & Hofmann, T. (2004). Non-redundant data clustering. *Proceedings ot the 4th IEEE International Conference on Data Mining*.

Guyon, I., Matic, N., & Vapnik, V. (1996). Discovering informative patterns and data cleaning. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy (Eds.), *Advances in knowledge discovery and data mining*, chapter 2, 181–204. Menlo Park, California: AAAI Press/The MIT Press.

Hand, D. (2002). Pattern detection and discovery. In D. Hand, N. Adams and R. Bolton (Eds.), *Pattern detection and discovery*. Springer.

Jennings, D., Amabile, T., & Ros, L. (1982). Informal covariation assessments: Data-based versus theory-based judgements. In D. Kahnemann, P. Slovic and A. Tversky (Eds.), *Judgement under uncertainty: Heuristics and biases*, 211 – 230. Cambridge: Cambridge University Press.

Kohavi, R., & John, G. H. (1998). The wrapper approach. In H. Liu and H. Motoda (Eds.), *Feature extraction, construction, and selection: A data mining perspective*, 33–50. Kluwer.

Mierswa, I., & Morik, K. (2005). Automatic feature extraction for classifying audio data. *Machine Learning Journal*, *58*, 127–149.

Miller, G. (1956). The magical number seven, plus or minus two: Some limits to our capacity for processing information. *Psychol Rev*, *63*, 81 – 97.

Murphy, P. M., & Aha, D. W. (1994). UCI repository of machine learning databases.

Platt, J. (1999). *Advances in large margin classifiers*, chapter Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. MIT Press.

Rüping, S. (2005). Classification with local models. In K. Morik, J.-F. Boulicaut and A. Siebes (Eds.), *Proceedings of the dagstuhl workshop on detecting local patterns*, Lecture Notes in Computer Science. Springer. to appear.

Sommer, E. (1996). *Theory restructering: A perspective on design & maintenance of Knowledge Based Systems*. Doctoral dissertation, University of Dortmund.

Tishby, N., Pereira, F., & Bialek, W. (1999). The information bottleneck method. *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing* (pp. 368–377).

Vapnik, V. (1998). *Statistical learning theory*. Chichester, GB: Wiley.

Invited Talk: Comparability and Semantics in Mining Multi-Represented Objects

Matthias Schubert SCHUBERT@DBS.IFI.LMU.DE
Institute for Informatics, University of Munich, D- 80538 Munich, Germany

Abstract

In recent years, the complexity of data objects in data mining applications has increased as well as their plain numbers. As a result the characteristics that are used to describe a data object are very heterogenous and often model different aspects of the same object. These different views on a certain data object has led to the development for various feature transformations and thus to multiple object representations. For example, a protein can be described by a text annotation, its amino acid sequence and its three dimensional structure. Thus, each protein can be described by three different feature vectors belonging to three different feature spaces. Another example are earth observation satellites taking images of the same area in multiple color spectra. To exploit these multiple representation for data mining, the solutions have to cope with two problems: Comparability and Semantics.

Comparability is a problem because the meaning of a pattern derived in one representation has to be related to the patterns derived from other representations in an unbiased way. For example, consider the distance between two objects that are given by two representations. Simply adding the distances in both representation might offer a very biased comparison if distances in the first representation tend to be much larger than in the second representation. Thus, we have to find a weighting that combines the distances in a fair way.

The other aspect is semantics describing the connection of a pattern in a single representation to the global patterns observed using all representations. For example, consider two objects that are found in the same cluster in two representations but are placed in different clusters in a third representation. The question if we should place an object in the same global cluster is depended on the meaning or semantics of the single representations. If it is enough that a certain object is similar with respect to one representation, we would place the objects in the same cluster in the final clustering based on all representations. However, in other applications similarity in one representation is only a hint that two objects are really similar. Thus, we have to demand that two object might be similar in more than one representation when placing them into the same cluster.

In this talk, we will discuss both problems and describe general directions for their solutions. Additionally, two methods using these approaches are introduced. The first is a method for kNN classification on multi-represented data objects. The idea of this method is to combine the decision sets of kNN classification in each representation. The second technique is a method for density-based clustering on multi-represented data objects. The idea of this method is to redefine the core-object predicate of the algorithm DBSCAN following varying semantics. Finally, we will provide the results of our experimental evaluation on a set of real world test sets from bio informatics and image processing. The results demonstrate that data mining algorithms following the mentioned approaches could significantly improve the quality of the derived patterns.

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

A Co-Regularization Approach to Semi-supervised Learning with Multiple Views

Vikas Sindhwani VIKASS@CS.UCHICAGO.EDU
Partha Niyogi NIYOGI@CS.UCHICAGO.EDU
Mikhail Belkin MISHA@CS.UCHICAGO.EDU
Department of Computer Science, University of Chicago, Chicago, IL 60637

Abstract

The Co-Training algorithm uses unlabeled examples in multiple views to bootstrap classifiers in each view, typically in a greedy manner, and operating under assumptions of view-independence and compatibility. In this paper, we propose a Co-Regularization framework where classifiers are learnt in each view through forms of multi-view regularization. We propose algorithms within this framework that are based on optimizing measures of agreement and smoothness over labeled and unlabeled examples. These algorithms naturally extend standard regularization methods like Support Vector Machines (SVM) and Regularized Least squares (RLS) for multi-view semi-supervised learning, and inherit their benefits and applicability to high-dimensional classification problems. An empirical investigation is presented that confirms the promise of this approach.

1. Introduction

A striking aspect of natural learning is the ability to integrate and process multi-modal sensory information with very little supervisory feedback. The scarcity of labeled examples, abundance of unlabeled data and presence of multiple representations are aspects of several applications of machine learning as well. An example is hypertext classification: Modern search engines can index more than a billion web-pages in a single web-crawl, but only a few can be hand-labeled and assembled into web directories. Each web-page has disparate descriptions: textual content, inbound and outbound hyperlinks, site and directory names, etc. Although traditional machine learning has focussed on two extremes of an information spectrum (supervised and unsupervised learning), a number of recent efforts have considered the middle-ground of semi-supervised learning, with or without a multi-view component (Belkin, Matveeva, & Niyogi, 2004; Belkin, Niyogi & Sindhwani, 2004; Sindhwani, Niyogi & Belkin, 2005; Joachims, 1999; Joachims, 2003; Blum & Mitchell, 1998; Brefeld & Scheffer; Chapelle & Zien, 2005; Zhou et al, 2004).

The Co-Training framework proposed in (Blum & Mitchell, 1998) has been among the first efforts that provided a widely successful algorithm with theoretical justifications. The framework employs two assumptions that allow unlabeled examples in multiple-views to be utilized effectively: (a) the assumption that the target functions in each view agree on labels of most examples (*compatibility* assumption) and (b) the assumption that the views are independent given the class label (*independence* assumption). The first assumption allows the complexity of the learning problem to be reduced by the constraint of searching over compatible functions; and the second assumption allows high performance to be achieved since it becomes unlikely for compatible classifiers trained on independent views to agree on an incorrect label. The co-training idea has become synonymous with a greedy agreement-maximization algorithm that is initialized by supervised classifiers in each view and then iteratively re-trained on boosted labeled sets, based on high-confidence predictions on the unlabeled examples. The original implementation in (Blum & Mitchell, 1998) runs this algorithm on naive-bayes classifiers defined in each view. For more on agreement maximization principles, see (Abney, 2002; Dasgupta, Littman & McAllester, 2001; Collins & Singer, 1999; Yarowsky, 1995).

In this paper, we present a *Co-Regularization* framework for multi-view semi-supervised learning. Our approach is based on implementing forms of multi-view

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

regularization using unlabeled examples. We suggest a family of algorithms within this framework: The Co-Regularized Least Squares (Co-RLS) algorithm performs a joint regularization that attempts to minimize disagreement in a least squared sense; the Co-Regularized Laplacian SVM and Least Squares (Co-LapSVM, Co-LapRLS) algorithms utilize multi-view graph regularizers to enforce complementary and robust notions of smoothness in each view. The recently proposed Manifold Regularization techniques (Belkin, Niyogi & Sindhwani, 2004; Sindhwani, 2004; Sindhwani, Niyogi & Belkin, 2005) are employed for Co-LapSVM and Co-LapRLS. Learning is performed by effectively exploiting useful structures collectively revealed with multiple representations.

We highlight features of the proposed algorithms:

1. These algorithms arise from natural extensions of the classical framework of regularization in Reproducing Kernel Hilbert Spaces. The unlabeled data is incorporated via additional regularizers that are motivated from recognized principles of semi-supervised learning.

2. The algorithms are non-greedy, involve convex cost functions and can be easily implemented.

3. The influence of unlabeled data and multiple views can be controlled explicitly. In particular, single view semi-supervised learning and standard supervised algorithms are special cases of this framework.

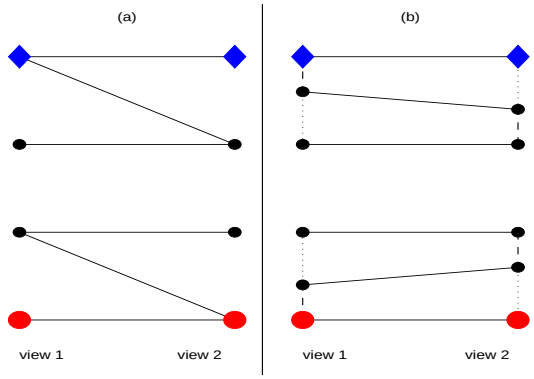
4. Experimental results demonstrate that the proposed methods out-perform standard co-training on synthetic and hypertext classification datasets.

In section 2, we setup the problem of semi-supervised learning in multiple views. In subsequent sections, we discuss the Co-Regularization framework, propose our algorithms and evaluate their empirical performance.

2. Multi-View Learning

In the multi-view semi-supervised learning setting, we have labeled examples $\{(x_i, y_i)\}_{i=1}^{l}$ and unlabeled examples $\{x_i\}_{l+1}^{l+u}$ where each example $x = (x^{(1)}, x^{(2)})$ is seen in two views with $x^{(1)} \in X^{(1)}$ and $x^{(2)} \in X^{(2)}$. The setup and the algorithms we discuss can also be generalized to more than two views. For the rest of this discussion, we consider binary classification problems where $y_i \in \{-1, 1\}$. The goal is to learn the function pair $f = (f^{(1)}, f^{(2)})$, where $f^{(1)} : X^{(1)} \mapsto \{-1, 1\}$ and $f^{(2)} : X^{(2)} \mapsto \{-1, 1\}$ are classifiers in the two views. In this paper, we will focus on how the availability of unlabeled examples and multiple views may be profitably leveraged for learning high-performance classifiers $f^{(1)}, f^{(2)}$ in each view.

Figure 1. Bipartite Graph Representation of multi-view learning. The small black circles are unlabeled examples.



How can unlabeled data and its multiple views help? In Figure 1(a), we reproduce the bipartite graph representation of the co-training setting, to initiate a discussion. The figure shows the two views of labeled and unlabeled examples, arranged as a bipartite graph. The left and right nodes in the graph are examples as seen in view 1 and view 2 respectively, with edges connecting the two views of an example. The unlabeled examples are shown as small black circles and the other examples are labeled. The class of compatible pairs of functions identically label two nodes in the same connected component of this graph. This may be interpreted as a requirement of smoothness over the graph for the pair $(f^{(1)}, f^{(2)})$. Thus, unlabeled examples provide empirical estimates of regularizers or measures of smoothness to enforce the right complexity for the pair $(f^{(1)}, f^{(2)})$.

In many applications, it is unrealistic for two examples to share a view exactly. A more realistic situation is depicted in Figure 1(b) where three types of edges are shown: (solid) edges connecting views of each example as in Figure 1(a); (dashed) edges connecting similar examples in each view; and (dotted) edges connecting examples in each view based on similarity in the other view. The similarity structure in one view induces a complementary notion of similarity in the other views with respect to which regularizers can be constructed using unlabeled data.

In the next section, we describe algorithms that arise from constructions of such regularizers.

3. Co-Regularization

The classical regularization framework (Poggio & Girosi, 1990; Schoelkopf & Smola, 2002; Vapnik, 1998) for supervised learning solves the following minimization problem :

$$f^* = \underset{f \in \mathcal{H}_K}{\operatorname{argmin}} \frac{1}{l} \sum_{i=1}^{l} V(x_i, y_i, f) + \gamma \|f\|_K^2 \qquad (1)$$

where $\mathcal{H}_K$ is an Reproducing Kernel Hilbert space (RKHS) of functions with kernel function K; $\{(x_i, y_i)\}_{i=1}^{l}$, is the labeled training set; and V is some loss function, such as squared loss for Regularized Least Squares (RLS) or the hinge loss function for Support Vector Machines (SVM). By the Representer theorem, the minimizer is a linear combination of kernel functions centered on the data:

$$f(x) = \sum_{i=1}^{l} \alpha_i K(x, x_i)$$

This real-valued function is thresholded and used for binary classification.

In the Co-regularization framework, we attempt to learn the pair $f = (f^{(1)}, f^{(2)})$ in a cross-product of two RKHS defined over the two views, i.e., $f^{(1)} \in \mathcal{H}_{K^{(1)}}$ and $f^{(2)} \in \mathcal{H}_{K^{(2)}}$. The key issue is imposing an appropriate notion of complexity on this pair so that a regularized solution effectively utilizes unlabeled data in the two views. We now describe some ideas.

Co-Regularized Least Squares

A natural idea is to attempt to learn the pair $f = (f^{(1)}, f^{(2)})$ so that each function correctly classifies the labeled examples, and the outputs of the pair agree over unlabeled examples. This suggests the following objective function:

$$(f^{(1)*}, f^{(2)*}) = \underset{\substack{f^{(1)} \in \mathcal{H}_{K^{(1)}} \\ f^{(2)} \in \mathcal{H}_{K^{(2)}}}}{\operatorname{argmin}} \sum_{i=1}^{l} \left[y_i - f^{(1)}(x_i^{(1)}) \right]^2 +$$

$$\mu \sum_{i=1}^{l} \left[y_i - f^{(2)}(x_i^{(2)}) \right]^2 + \gamma_1 \|f^{(1)}\|^2_{\mathcal{H}_{K^{(1)}}} +$$

$$\gamma_2 \|f^{(2)}\|^2_{\mathcal{H}_{K^{(2)}}} + \frac{\gamma_C}{(l+u)} \sum_{i=1}^{l+u} \left[f^{(1)}(x_i^{(1)}) - f^{(2)}(x_i^{(2)}) \right]^2$$

Here, μ is a real-valued parameter to balance data fitting in the two views, γ_1, γ_2 are regularization parameters for the RKHS norms in the two views, and γ_C is the coupling parameter that regularizes the pair towards compatibility using unlabeled data. It is easy to see that a representer theorem holds that expresses the minimizing pair $\left(f^{(1)*}(x^{(1)}), f^{(2)*}(x^{(2)})\right)$ in the following form:

$$\left(\sum_{i=1}^{l+u} \alpha_i K^{(1)}(x^{(1)}, x_i^{(1)}) \quad , \quad \sum_{i=1}^{l+u} \beta_i K^{(2)}(x^{(2)}, x_i^{(2)}) \right)$$

The $(l+u)$ dimensional expansion coefficient vectors α, β may be computed by solving the following coupled linear system:

$$\left[\frac{1}{l} J K_1 + \gamma_1 I + \frac{\gamma_C}{l+u} K_1 \right] \alpha - \frac{\gamma_C}{l+u} K_2 \beta = \frac{1}{l} Y$$

$$\left[\frac{\mu}{l} J K_2 + \gamma_2 I + \frac{\gamma_C}{l+u} K_2 \right] \beta - \frac{\gamma_C}{l+u} K_1 \alpha = \frac{\mu}{l} Y$$

where Y is a label vector given by $Y_i = y_i$ for $1 \le i \le l$ and $Y_i = 0$ for $l+1 \le i \le l+u$; J is a diagonal matrix given by $J_{ii} = |Y_i|$, and K_1, K_2 are gram matrices of the kernel functions $K^{(1)}, K^{(2)}$ over labeled and unlabeled examples.

When $\gamma_C = 0$, the system ignores unlabeled data and yields an uncoupled pair of solutions corresponding to supervised RLS. We also note a curious relationship over coefficients corresponding to unlabeled examples: $\gamma_1 \alpha_i = -\gamma_2 \beta_i$ for $l+1 \le i \le l+u$. The algorithm appears to work well in practice when orthogonality to the constant function is enforced over the data to avoid all unlabeled examples from being identically classified.

Working with the hinge loss, one can also extend SVMs in a similar manner. This has not been attempted in this paper.

Co-Laplacian RLS and Co-Laplacian SVM

The intuitions from the discussion concerning Figure 1(b) is to learn the pair $f = (f^{(1)}, f^{(2)})$ so that each function correctly classifies the labeled examples and is smooth with respect to similarity structures in both views. These structures may be encoded as graphs on which regularization operators may be defined and then combined to form a multi-view regularizer. The function pair is indirectly coupled through this regularizer.

We assume that for each view (indexed by $s = 1, 2$), we can construct a similarity graph whose adjacency matrix is $W^{(s)}$, where $W_{ij}^{(s)}$ measures similarity between $x_i^{(s)}$ and $x_j^{(s)}$. The Laplacian matrix of this graph is defined as $L^{(s)} = D^{(s)} - W^{(s)}$ where $D^{(s)}$ is the diagonal degree matrix $D_{ii}^{(s)} = \sum_j W_{ij}^{(s)}$. The graph Laplacian is a positive semi-definite operator on functions defined over vertices of the graph. It provides

the following smoothness functional on the graph:

$$\mathbf{g}^T L^{(s)} \mathbf{g} = \sum_{ij} (g_i - g_j)^2 W_{ij}^{(s)}$$

where $\mathbf{g}$ is a vector identifying a function on the graph whose value is g_i on node i. Other regularization operators can also be defined using the graph Laplacian (Kondor & Lafferty, 2003; Smola & Kondor, 2003; Belkin, Matveeva, & Niyogi, 2004).

One way to construct a multi-view regularizer is to simply take a convex combination $L = (1-\alpha)L^{(1)} + \alpha L^{(2)}$ where $\alpha \geq 0$ is a non-negative parameter which controls the influence of the two views. To learn the pair $f = (f^{(1)*}, f^{(2)*})$, we solve the following optimization problems for $s = 1, 2$ using squared loss or hinge loss:

$$f^{(s)*} = \underset{f^{(s)} \in \mathcal{H}_{K^{(s)}}}{\operatorname{argmin}} \frac{1}{l} \sum_{i=1}^{l} V(x_i^{(s)}, y_i, f^{(s)}) + \gamma_A^{(s)} \|f^{(s)}\|^2_{K^{(s)}} + \gamma_I^{(s)} \mathbf{f}^{(s)T} L \mathbf{f}^{(s)}$$

where $\mathbf{f}^{(s)}$ denotes the vector $\left(f^{(s)}(x_1^{(s)}), \ldots, f^{(s)}(x_{l+u}^{(s)})\right)^T$; and the regularization parameters $\gamma_A^{(s)}, \gamma_I^{(s)}$ control the influence of unlabeled examples relative to the RKHS norm.

The solutions to these optimization problems produce the recently proposed Laplacian SVM (for hinge loss) or Laplacian RLS (for squared loss) classifiers trained with the multi-view graph regularizer (Belkin, Niyogi & Sindhwani, 2004; Sindhwani, Niyogi & Belkin, 2005; Sindhwani, 2004). The resulting algorithms are termed Co-Laplacian SVM and Co-Laplacian RLS respectively.

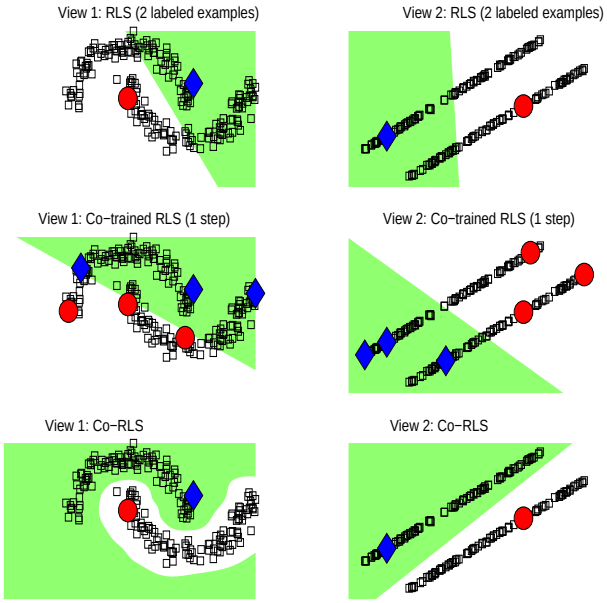
The solutions are obtained by training a standard SVM or RLS using the following modified kernel function:

$$\tilde{K}^{(s)}(x^{(s)}, z^{(s)}) = K^{(s)}(x^{(s)}, z^{(s)}) - \mathbf{k}^{\mathbf{T}}_{\mathbf{x}^{(s)}} (I + MG^{(s)})^{-1} M \mathbf{k}_{\mathbf{z}^{(s)}}$$

where $G^{(s)}$ is the gram matrix of the kernel function $K^{(s)}$; $\mathbf{k}_{\mathbf{x}^{(s)}}$ denotes the vector $\left(K^{(s)}(x_1^{(s)}, x^{(s)}), \ldots, K^{(s)}(x_n^{(s)}, x^{(s)})\right)^T$ and $M = \frac{\gamma_I^{(s)}}{\gamma_A^{(s)}} L$. See (Sindhwani, Niyogi & Belkin, 2005) for a derivation of this kernel.

When $\alpha = 0$ for view 1 or $\alpha = 1$ for view 2, the multi-view aspect is ignored and the pair consists of Laplacian SVM or Laplacian RLS in each view. When $\gamma_I = 0$, the unlabeled data is ignored and the pair consists of standard SVM or RLS classifiers.

Figure 2. Two-Moons-Two-Lines : RLS, Co-trained RLS and Co-RLS



The idea of combining graph regularizers and its connection to co-training has been briefly discussed in (Joachims, 2003) in the context of applying spectral graph transduction (SGT) in multi-view settings. However, unlike co-training, SGT does not produce classifiers defined everywhere in $X^{(1)}, X^{(2)}$ so that predictions cannot be made on novel test points. By optimizing in reproducing kernel Hilbert spaces defined everywhere, Co-Laplacian SVM and RLS can also extend beyond the unlabeled examples.

4. Experiments

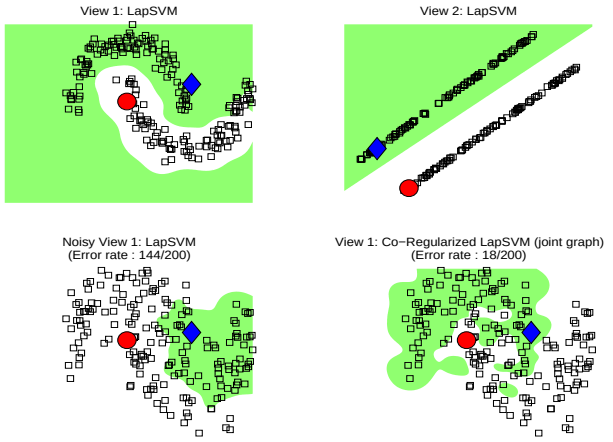
We performed experiments on a toy multi-view dataset and a hypertext document categorization task.

Two-Moons-Two-Lines Toy Example

Figure 2 and Figure 3 demonstrate Co-Regularization ideas on a toy dataset in which objects in two classes appear as two moons in one view and two oriented lines in another. Class conditional view independence is enforced by randomly associating points on one moon with points on one line, somewhat like the News 2×2 dataset in (Nigam & Ghani 2000). One example is labeled from each class and shown as the large colored diamond and circle; the other examples are unlabeled. We chose a Gaussian kernel for the two moons view and a linear kernel for the two lines view.

In the top panel of Figure 2, we see that a supervised Regularized least squares classifier is unable to

Figure 3. Two-Moons-Two-Lines : Laplacian SVM and Co-Laplacian SVM



produce reasonable classifiers with only 2 labeled examples. In the middle panel, we add two more labeled examples based on the most confident predictions (which are actually incorrect) of the supervised classifiers on the unlabeled data. The middle panel shows the classifiers obtained after 1 iteration of standard co-training with the boosted set of 4 labeled examples. Since greedy co-training does not revise conjectured labels, subsequent training fails to yield good classifiers in either view. By contrast, Co-Regularized Least squares classifiers, shown in panel 3, effectively use the unlabeled data in two views.

In the top panel of Figure 3, we show single-view semi-supervised learning with Laplacian SVMs in the two views. We then add noise to the two-moons view so that the two clusters are merged. This is shown in the bottom left panel. In this case, the unlabeled data fails to provide any structure for Laplacian SVM to exploit. However, when the joint graph laplacian is used, the rich structure in the two-lines view can be used to recover good decision boundaries in the two moons view. The bottom right panel shows the boundaries constructed by Co-Laplacian SVM.

Hypertext Categorization

We considered the WebKB hypertext categorization task studied in (Blum & Mitchell, 1998; Joachims, 2003; Nigam & Ghani 2000). There are 1051 web documents belonging to two classes: *course* or *non-course* from four universities. Only 12 examples are labeled. The two views are the textual content of a webpage (which we will call *page* representation) and the anchortext on links on other webpages pointing to the webpage (*link* representation).

The data was preprocessed into 3000 features for the page-view and 1840 features for the link view using the Rainbow software (McAllum, 1996). We used linear kernels for both views. We also considered a page+link representation with concatenated features.

The performance of several methods as measured by mean precision-recall breakeven point (PRBEP) is tabulated in Table 1. These methods are (a) RLS, SVM on fully labeled data sets and with 12 randomly chosen labeled examples; (b) single-view semi-supervised methods: SGT (Joachims, 2003), TSVM (Joachims, 1999), Laplacian SVM, Laplacian RLS (Belkin, Niyogi & Sindhwani, 2004; Sindhwani, Niyogi & Belkin, 2005); (c) multi-view semi-supervised methods: Co-RLS, Co-trained RLS, Co-trained SVM, Co-LapRLS and Co-LapSVM. In Table 1, Co-LapRLS1, Co-LapSVM1 use $\alpha = 0.5$ to combine graph Laplacians in page and link views; and Co-LapRLS2, Co-LapSVM2 use the mean graph Laplacian over page, link and page+link views, to bias classifiers in each view. The performance of supervised classifiers with full labels (RLS (full) and SVM (full)) is the mean PRBEP for 10-fold cross-validation. For all other methods, we average over random choices of 12 labeled examples (making sure that each class is sampled at least once) and measure the mean PRBEP evaluated over the remaining 1039 examples. We avoided the model selection issue due to the small size of the labeled set and chose best parameters over a small range of values.

The results in table 1 suggest that Co-LapSVM and Co-LapRLS are able to effectively use unlabeled examples in the two views. The link and page classifiers using 12 labeled examples, 1039 unlabeled examples and multi-view regularizers match the performance of supervised classifiers with access to all the labels. We also see that Co-RLS outperforms Co-trained RLS. In Table 2, we report the performance of Co-Laplacian SVM (using the mean graph Laplacian over the page, link and page+link views) in classifying unlabeled and test web-documents of four universities. The high correlation between performance on unlabeled and unseen test examples suggests that the method provides good extension outside the training set.

5. Conclusion

We have proposed extensions of regularization algorithms in a setting where unlabeled examples are easily available in multiple views. The algorithms provide natural extensions for SVM and RLS in such settings. We plan to further investigate the properties of these algorithms and benchmark them on real world tasks.

Table 1. Mean precision-recall breakeven points over unlabeled documents for a hypertext classification task.

View → Classifier ↓	link	page	page+ link
RLS (full)	94.4	94.0	97.8
SVM (full)	93.7	93.5	99.0
RLS (12)	72.0	71.6	78.3
SVM (12)	74.4	77.8	84.4
SGT	78.0	89.3	93.4
TSVM	85.5	91.4	92.2
LapRLS	80.8	89.0	93.1
LapSVM	81.9	89.5	93.6
Co-trained RLS	74.8	80.2	-
Co-RLS	80.8	90.1	-
Co-LapRLS1	93.1	90.8	90.4
Co-LapRLS2	94.4	92.0	93.6
Co-trained SVM	88.3	88.7	-
Co-LapSVM1	93.2	93.2	90.8
Co-LapSVM2	94.3	93.3	94.2

Table 2. Mean precision-recall breakeven points over test documents and over unlabeled documents (test , unlabeled)

University → View ↓	page+link	page	link
Cornell	91.6 , 90.9	88.9 , 88.8	88.2 , 88.7
Texas	94.8 , 95.5	91.6 , 92.4	90.9 , 93.5
Washington	94.7 , 94.9	94.0 , 93.9	93.7 , 92.4
Wisconsin	92.0 , 91.4	87.6 , 86.6	86.1 , 84.5

References

Abney, S. (2002) *Bootstrapping.* Proceedings of ACL 40

Blum A. & Mitchell T. (1998) *Combining Labeled and Unlabeled Data with Co-Training.* COLT

Belkin M., Niyogi P. & Sindhwani V. (2004) *Manifold Regularization : A Geometric Framework for Learning for Examples.* Technical Report, Dept. of Computer Science, Univ. of Chicago, TR-2004-06

Belkin M., Matveeva I. & Niyogi P. (2004) *Regression and Regularization on Large Graphs.* COLT

Brefeld, U. & Scheffer, T. (2004) *Co-EM Support Vector Learning.* ICML

Chapelle O. & Zien,A. (2005) *Semi-Supervised Classification by Low Density Separation.* Artificial Intelligence and Statistics, 2005

Collins, M. & Singer, Y. (1999) *Unsupervised Models for Named Entity Classification.* EMNLP/VLC-99

Dasgupta, S., Littman, M., & McAllester, D. (2001) *PAC Generalization Bounds for Co-Training* NIPS

Joachims T. (1999) *Transductive Inference for Text Classification using Support Vector Machines.* ICML

Joachims T. (2003) *Transductive Learning via Spectral Graph Partitioning.* ICML

Kondor I.R.& Lafferty, J. (2003) *Diffusion Kernels on Graphs and Other Discrete Input Spaces.* ICML

Nigam, K. & Ghani, R. (2001) *Kamal Nigam and Rayid Ghani. Analyzing the Effectiveness and Applicability of Co-training.* CIKM, pp. 86-93

McCallum, A.K. (1996) *Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering.* http://www.cs.cmu.edu/∼mccallum/bow

Poggio, T., & Girosi (1990) *Regularization Algorithms for Learning That Are Equivalent to Multilayer Networks.* Science 247:978-982

Schoelkopf, B. & Smola, A.J. (2002) *Learning with Kernels.* MIT Press, Cambridge, MA

Sindhwani, V. (2004) *Kernel Machines for Semi-supervised Learning.* Masters Thesis, University of Chicago

Sindhwani, V., Niyogi, P., & Belkin, M. (2005) *Beyond the point cloud: from Transductive to Semi-supervised Learning.* ICML

Smola A.J., & Kondor I.R (2003) *Kernels and Regularization on Graphs.* COLT

Vapnik V, (1998) *Statistical Learning Theory.* Wiley-Interscience

Yarowsky, D. (1995) *Unsupervised Word Sense Disambiguation Rivaling Supervised Methods.* ACL

Zhou, D., Bousquet, O., Lal, T. N., Weston J., & Schoelkopf, B. (2004) *Learning with Local and Global Consistency.* NIPS 16

Analytical Kernel Matrix Completion with Incomplete Multi-View Data

David Williams DPW@EE.DUKE.EDU
Lawrence Carin LCARIN@EE.DUKE.EDU
Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA

Abstract

In multi-view remote sensing applications, incomplete data can result when only a subset of sensors are deployed at certain regions. We derive a closed-form expression for computing a Gaussian kernel when faced with incomplete data. This expression is obtained by analytically integrating out the missing data. This result can subsequently be used in conjunction with any kernel-based classifier. The superiority of the proposed method over two common imputation schemes is demonstrated on one benchmark data set and three real (measured) multi-view land mine data sets.

1. Introduction

The incomplete-data problem, in which certain features are missing from particular feature vectors, exists in a wide range of fields, including social sciences, computer vision, biological systems, and remote sensing. For example, partial responses in surveys are common in the social sciences, leading to incomplete data sets with arbitrary patterns of missing data. In multi-view remote sensing applications, incomplete data can result when only a subset of sensors (*e.g.,* radar, infrared, acoustic) are deployed at certain regions. Increasing focus in the future on using (and fusing data from) multiple sensors, information sources, or "views" will make such incomplete data problems more common (see (Tsuda, Akaho & Asai, 2003; Lanckriet et al., 2004)).

Incomplete data problems are often circumvented in the initial stage of analysis—before specific algorithms become involved—via imputation (*i.e.,* by "completing" the missing data by filling in specific values). Common imputation schemes include "completing" missing data with zeros, the unconditional mean, or the conditional mean (if one has an estimate for the distribution of missing features given the observed features, $p\left(\boldsymbol{x}_i^{m_i}|\boldsymbol{x}_i^{o_i}\right)$).

When kernel methods such as the SVM (Schölkopf & Smola, 2002) are employed, one can either first complete the data and then compute the kernel matrix, or else complete and compute the kernel matrix in a single stage. Semidefinite programming has been used to complete kernel matrices that have only a *limited* number of missing elements (Graepel, 2002). The *em* algorithm (Tsuda, Akaho & Asai, 2003) is applicable when both an incomplete auxiliary kernel matrix and a complete primary kernel matrix exist, but not when the patterns of missing data are completely arbitrary. This assumption may be tolerable in certain applications, but it is not appropriate for the general missing data problem.

By making only two assumptions—that $p\left(\boldsymbol{x}_i^{m_i}|\boldsymbol{x}_i^{o_i}\right)$ is a Gaussian mixture model (GMM), and that a Gaussian kernel is employed—we can analytically calculate the kernel matrix from incomplete data by integrating out the missing data. The first assumption is mild, since it is well-known that a mixture of Gaussians can approximate any distribution. The second assumption is not overly limiting as the Gaussian kernel is one of the most commonly used kernel forms. In fact, if one assumes a linear or polynomial kernel instead of a Gaussian kernel, the missing data of the kernel matrix can still be analytically integrated out. However, the calculations for these kernels are trivial, so we focus here on the more interesting case of the Gaussian kernel. After obtaining the kernel matrix, any kernel-based method may be employed, as would be done for an ordinary complete-data problem.

This paper is organized as follows. In Section 2, we derive the expression to analytically compute a kernel

Appearing in *Proceedings of the Workshop on Learning with Multiple Views*, 22^{nd} ICML, Bonn, Germany, 2005.

matrix in the presence of incomplete data for Gaussian kernels. Experimental classification results on one benchmark machine learning data set and on three real multi-view land mine data sets are shown in Section 3, before concluding remarks are made in Section 4.

2. Kernel Matrix with Incomplete Data

A data point $\boldsymbol{x}_i$ may be mapped into feature space via the positive semidefinite kernel function K. Computing the kernel for every pair of data points results in the symmetric, positive semidefinite kernel matrix $\mathbf{K}$. The ij-th entry of this kernel matrix, K_{ij}, is a measure of similarity between two data points, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Our goal is to obtain the kernel matrix when incomplete data exists. We solve this task in a multi-view framework, treating incomplete data as the result of only a subset of views being observed for any given data point. However, this framework is not limiting because one can simply treat each individual feature as coming from a unique view.

2.1. Derivation of the Kernel Matrix

Let $\boldsymbol{x}_i^s$ denote the data (features) of the i-th data point from the set of views s. Let o_i and m_i denote the sets of observed views and missing views for data point $\boldsymbol{x}_i$, respectively. The notation $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ indicates that $\boldsymbol{x}$ is distributed as a Gaussian with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. Using all available data, we model the joint probability $p\left(\boldsymbol{x}_i^{m_i}, \boldsymbol{x}_i^{o_i}\right)$ using a (Z-component) Gaussian mixture model

$$\begin{aligned} p\left(\boldsymbol{x}_i^{m_i}, \boldsymbol{x}_i^{o_i}\right) &= \sum_{\zeta=1}^{Z} \varpi_\zeta \mathcal{N}\left(\boldsymbol{x}_i | \boldsymbol{\theta}_\zeta, \boldsymbol{\Theta}_\zeta\right) \\ &= \sum_{\zeta=1}^{Z} \varpi_\zeta \mathcal{N}\left(\left[\begin{array}{c} \boldsymbol{x}_i^{m_i} \\ \boldsymbol{x}_i^{o_i} \end{array}\right] \middle| \left[\begin{array}{c} \boldsymbol{\theta}_\zeta^{m_i} \\ \boldsymbol{\theta}_\zeta^{o_i} \end{array}\right], \right. \\ &\qquad \left. \left[\begin{array}{cc} \boldsymbol{\Theta}_\zeta^{[m_i m_i]} & \boldsymbol{\Theta}_\zeta^{[m_i o_i]} \\ \left(\boldsymbol{\Theta}_\zeta^{[m_i o_i]}\right)^T & \boldsymbol{\Theta}_\zeta^{[o_i o_i]} \end{array}\right]\right) \end{aligned} \quad (1)$$

where ϖ_ζ are the non-negative mixing coefficients that sum to unity.

Any conditional distribution derived from this joint probability will also be a mixture of Gaussians. Specifically,

$$p\left(\boldsymbol{x}_i^{m_i} \middle| \boldsymbol{x}_i^{o_i}\right) = \sum_{\zeta=1}^{Z} \pi_\zeta^i \mathcal{N}\left(\boldsymbol{x}_i^{m_i} \middle| \boldsymbol{\mu}_\zeta^{m_i}, \boldsymbol{\Sigma}_\zeta^{m_i}\right) \quad (2)$$

where

$$\begin{aligned} \pi_\zeta^i &= \frac{\varpi_\zeta \mathcal{N}\left(\boldsymbol{x}_i^{o_i} \middle| \boldsymbol{\theta}_\zeta^{o_i}, \boldsymbol{\Theta}_\zeta^{[o_i o_i]}\right)}{\sum_{\xi=1}^{Z} \varpi_\xi \mathcal{N}\left(\boldsymbol{x}_i^{o_i} \middle| \boldsymbol{\theta}_\xi^{o_i}, \boldsymbol{\Theta}_\xi^{[o_i o_i]}\right)} \\ \boldsymbol{\mu}_\zeta^{m_i} &= \boldsymbol{\theta}_\zeta^{m_i} + \boldsymbol{\Omega}\left(\boldsymbol{x}_i^{o_i} - \boldsymbol{\theta}_\zeta^{o_i}\right) \\ \boldsymbol{\Sigma}_\zeta^{m_i} &= \boldsymbol{\Theta}_\zeta^{[m_i m_i]} - \boldsymbol{\Omega}\left(\boldsymbol{\Theta}_\zeta^{[m_i o_i]}\right)^T \\ \boldsymbol{\Omega} &\equiv \boldsymbol{\Theta}_\zeta^{[m_i o_i]}\left(\boldsymbol{\Theta}_\zeta^{[o_i o_i]}\right)^{-1}. \end{aligned}$$

We also employ a Gaussian kernel function

$$\begin{aligned} K_{ij} = K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) &= z_\kappa \cdot \exp\left\{\frac{||\boldsymbol{x}_j - \boldsymbol{x}_i||_2^2}{-2\sigma_\kappa^2}\right\} \\ &= z_\kappa \cdot \exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_j - \boldsymbol{x}_i\right)^T \boldsymbol{\Sigma}_\kappa^{-1}\left(\boldsymbol{x}_j - \boldsymbol{x}_i\right)\right\} \end{aligned} \quad (3)$$

where $\boldsymbol{\Sigma}_\kappa = \mathrm{diag}\left(\sigma_\kappa^2\right)$ and $z_\kappa = (2\pi)^{-d/2}\left|\boldsymbol{\Sigma}_\kappa\right|^{-1/2}$. For $S > 1$ views, this kernel can be written as a product of the individual-view kernels in various forms:

$$\begin{aligned} K_{ij} &= \prod_{s=1}^{S} K_{ij}^s = \prod_{s=1}^{S} \mathcal{N}\left(\boldsymbol{x}_j^s \middle| \boldsymbol{x}_i^s, \boldsymbol{\Sigma}_\kappa^s\right) \\ &= \prod_{s=1}^{S} \mathcal{N}\left(\boldsymbol{x}_j^s - \boldsymbol{x}_i^s \middle| \boldsymbol{0}, \boldsymbol{\Sigma}_\kappa^s\right). \end{aligned}$$

In the following, we wish to derive the kernel K_{ij} between two arbitrary data points, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, with incomplete data. To do so, we will utilize the Gaussian mixture model of $p\left(\boldsymbol{x}_i^{m_i}, \boldsymbol{x}_i^{o_i}\right)$, which we assume has already been obtained. Because of the absence of data from some of the views, the incomplete data must be integrated out. For ease of reading, we give the complete derivation uninterrupted by text, opting to justify each step afterward. Note that $a \cap b$ indicates the intersection of sets a and b. The desired kernel is

$$\begin{aligned} & K\left(\boldsymbol{x}_i, \boldsymbol{x}_j \middle| \boldsymbol{x}_i^{o_i}, \boldsymbol{x}_j^{o_j}\right) \\ = & \int d\boldsymbol{x}_j^{m_j} \int d\boldsymbol{x}_i^{m_i} p\left(\boldsymbol{x}_i^{m_i}, \boldsymbol{x}_j^{m_j} \middle| \boldsymbol{x}_i^{o_i}, \boldsymbol{x}_j^{o_j}\right) \\ & K\left(\boldsymbol{x}_i, \boldsymbol{x}_j \middle| \boldsymbol{x}_i^{o_i}, \boldsymbol{x}_i^{m_i}, \boldsymbol{x}_j^{o_j}, \boldsymbol{x}_j^{m_j}\right) \\ \overset{(a)}{=} & \int d\boldsymbol{x}_j^{m_j} \int d\boldsymbol{x}_i^{m_i} p\left(\boldsymbol{x}_i^{m_i} \middle| \boldsymbol{x}_i^{o_i}\right) p\left(\boldsymbol{x}_j^{m_j} \middle| \boldsymbol{x}_j^{o_j}\right) \\ & K\left(\boldsymbol{x}_i, \boldsymbol{x}_j \middle| \boldsymbol{x}_i^{o_i}, \boldsymbol{x}_i^{m_i}, \boldsymbol{x}_j^{o_j}, \boldsymbol{x}_j^{m_j}\right) \\ \overset{(b)}{=} & K_{ij}^{o_i \cap o_j} \int d\boldsymbol{x}_j^{m_j} K_{ij}^{o_i \cap m_j} p\left(\boldsymbol{x}_j^{m_j} \middle| \boldsymbol{x}_j^{o_j}\right) \\ & \int d\boldsymbol{x}_i^{m_i} K_{ij}^{m_i} p\left(\boldsymbol{x}_i^{m_i} \middle| \boldsymbol{x}_i^{o_i}\right) \end{aligned}$$

$$
\begin{aligned}
&\overset{(c)}{=} && K_{ij}^{o_i\cap o_j}\int d\boldsymbol{x}_j^{m_j}\,K_{ij}^{o_i\cap m_j}\,p\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{x}_j^{o_j}\right)\\
&&&\int d\boldsymbol{x}_i^{m_i}\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_i^{m_i}\;\middle|\;\boldsymbol{\mu}_\zeta^{m_i},\boldsymbol{\Sigma}_\zeta^{m_i}\right)\\
&&&\mathcal{N}\left(\boldsymbol{x}_j^{m_i}-\boldsymbol{x}_i^{m_i}\;\middle|\;\mathbf{0},\boldsymbol{\Sigma}_\kappa^{m_i}\right)\\
&\overset{(d)}{=} && K_{ij}^{o_i\cap o_j}\int d\boldsymbol{x}_j^{m_j}\,K_{ij}^{o_i\cap m_j}\,p\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{x}_j^{o_j}\right)\\
&&&\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i}\;\middle|\;\boldsymbol{\mu}_\zeta^{m_i},\boldsymbol{\Sigma}_\kappa^{m_i}+\boldsymbol{\Sigma}_\zeta^{m_i}\right)\\
&\overset{(e)}{=} && K_{ij}^{o_i\cap o_j}\int d\boldsymbol{x}_j^{m_j}\,K_{ij}^{o_i\cap m_j}\,p\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{x}_j^{o_j}\right)\\
&&&\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap m_j}\;\middle|\;\boldsymbol{f},\mathbf{F}\right)\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap o_j}\;\middle|\;\boldsymbol{g},\mathbf{G}\right)\\
&\overset{(f)}{=} && K_{ij}^{o_i\cap o_j}\int d\boldsymbol{x}_j^{m_j}\,p\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{x}_j^{o_j}\right)\\
&&&\mathcal{N}\left(\boldsymbol{x}_j^{o_i\cap m_j}\;\middle|\;\boldsymbol{x}_i^{o_i\cap m_j},\boldsymbol{\Sigma}_\kappa^{o_i\cap m_j}\right)\\
&&&\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap m_j}\;\middle|\;\boldsymbol{f},\mathbf{F}\right)\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap o_j}\;\middle|\;\boldsymbol{g},\mathbf{G}\right)\\
&\overset{(g)}{=} && K_{ij}^{o_i\cap o_j}\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap o_j}\;\middle|\;\boldsymbol{g},\mathbf{G}\right)\\
&&&\int d\boldsymbol{x}_j^{m_j}\,p\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{x}_j^{o_j}\right)\mathcal{N}\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{a},\mathbf{A}\right)\\
&\overset{(h)}{=} && K_{ij}^{o_i\cap o_j}\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap o_j}\;\middle|\;\boldsymbol{g},\mathbf{G}\right)\\
&&&\int d\boldsymbol{x}_j^{m_j}\sum_{\xi=1}^{Z}\pi_\xi^j\mathcal{N}\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{b},\mathbf{B}\right)\mathcal{N}\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{a},\mathbf{A}\right)\\
&\overset{(i)}{=} && K_{ij}^{o_i\cap o_j}\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap o_j}\;\middle|\;\boldsymbol{g},\mathbf{G}\right)\\
&&&\sum_{\xi=1}^{Z}\pi_\xi^j\int d\boldsymbol{x}_j^{m_j}\,z_c\mathcal{N}\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{c},\mathbf{C}\right)\\
&\overset{(j)}{=} && K_{ij}^{o_i\cap o_j}\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap o_j}\;\middle|\;\boldsymbol{g},\mathbf{G}\right)\sum_{\xi=1}^{Z}\pi_\xi^j z_c.
\end{aligned}
\tag{4}
$$

In the derivation leading to (4), (a) follows because $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are independent; (b) follows by defining

$$
\begin{aligned}
K_{ij} &= K\left(\boldsymbol{x}_i,\boldsymbol{x}_j\;\middle|\;\boldsymbol{x}_i^{o_i},\boldsymbol{x}_i^{m_i},\boldsymbol{x}_j^{o_j},\boldsymbol{x}_j^{m_j}\right)\\
&= K_{ij}^{o_i}K_{ij}^{m_i}\\
&= K_{ij}^{o_i\cap m_j}K_{ij}^{o_i\cap o_j}K_{ij}^{m_i};
\end{aligned}
$$

(c) follows by writing

$$
\begin{aligned}
K_{ij}^{m_i} &= \mathcal{N}\left(\boldsymbol{x}_j^{m_i}-\boldsymbol{x}_i^{m_i}\;\middle|\;\mathbf{0},\boldsymbol{\Sigma}_\kappa^{m_i}\right)\\
&= \mathcal{N}\left(\left[\begin{array}{c}\boldsymbol{x}_j^{m_i\cap m_j}-\boldsymbol{x}_i^{m_i\cap m_j}\\ \boldsymbol{x}_j^{o_i\cap m_j}-\boldsymbol{x}_i^{o_i\cap m_j}\end{array}\right]\;\middle|\;\left[\begin{array}{c}\mathbf{0}\\ \mathbf{0}\end{array}\right],\right.\\
&\qquad\left.\left[\begin{array}{cc}\boldsymbol{\Sigma}_\kappa^{m_i\cap m_j} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{\Sigma}_\kappa^{m_i\cap o_j}\end{array}\right]\right)
\end{aligned}
$$

and

$$
\begin{aligned}
p\left(\boldsymbol{x}_i^{m_i}\;\middle|\;\boldsymbol{x}_i^{o_i}\right) &= \sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_i^{m_i}\;\middle|\;\boldsymbol{\mu}_\zeta^{m_i},\boldsymbol{\Sigma}_\zeta^{m_i}\right)\\
&= \sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\left[\begin{array}{c}\boldsymbol{x}_i^{m_i\cap m_j}\\ \boldsymbol{x}_i^{o_i\cap m_j}\end{array}\right]\;\middle|\;\left[\begin{array}{c}\boldsymbol{\mu}_\zeta^{m_i\cap m_j}\\ \boldsymbol{\mu}_\zeta^{m_i\cap o_j}\end{array}\right],\right.\\
&\qquad\left.\left[\begin{array}{cc}\boldsymbol{\Sigma}_\zeta^{m_i[m_jm_j]} & \boldsymbol{\Sigma}_\zeta^{m_i[m_jo_j]}\\ \left(\boldsymbol{\Sigma}_\zeta^{m_i[m_jo_j]}\right)^T & \boldsymbol{\Sigma}_\zeta^{m_i[o_jo_j]}\end{array}\right]\right);
\end{aligned}
$$

(d) follows because the right-most integral is a convolution of two Gaussians; (e) follows from conditioning on $\boldsymbol{x}_j^{o_i\cap m_j}$ so that

$$
\begin{aligned}
&\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i}\;\middle|\;\boldsymbol{\mu}_\zeta^{m_i},\boldsymbol{\Sigma}_\kappa^{m_i}+\boldsymbol{\Sigma}_\zeta^{m_i}\right)\\
&\quad=\sum_{\zeta=1}^{Z}\pi_\zeta^i\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap m_j}\;\middle|\;\boldsymbol{f},\mathbf{F}\right)\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap o_j}\;\middle|\;\boldsymbol{g},\mathbf{G}\right)
\end{aligned}
$$

where

$$
\begin{aligned}
\boldsymbol{f} &= \boldsymbol{\mu}_\zeta^{m_i\cap m_j}+\boldsymbol{\Upsilon}\left(\boldsymbol{x}_j^{m_i\cap o_j}-\boldsymbol{\mu}_\zeta^{m_i\cap o_j}\right)\\
\mathbf{F} &= \left(\boldsymbol{\Sigma}_\kappa^{m_i\cap m_j}+\boldsymbol{\Sigma}_\zeta^{m_i[m_jm_j]}\right)-\boldsymbol{\Upsilon}\left(\boldsymbol{\Sigma}_\zeta^{m_i[m_jo_j]}\right)^T\\
\boldsymbol{\Upsilon} &\equiv \boldsymbol{\Sigma}_\zeta^{m_i[m_jo_j]}\left(\boldsymbol{\Sigma}_\kappa^{m_i\cap o_j}+\boldsymbol{\Sigma}_\zeta^{m_i[o_jo_j]}\right)^{-1}\\
\boldsymbol{g} &= \boldsymbol{\mu}_\zeta^{m_i\cap o_j}\\
\mathbf{G} &= \boldsymbol{\Sigma}_\kappa^{m_i\cap o_j}+\boldsymbol{\Sigma}_\zeta^{m_j[o_jo_j]};
\end{aligned}
$$

(f) follows because

$$
K_{ij}^{o_i\cap m_j}=\mathcal{N}\left(\boldsymbol{x}_j^{o_i\cap m_j}\;\middle|\;\boldsymbol{x}_i^{o_i\cap m_j},\boldsymbol{\Sigma}_\kappa^{o_i\cap m_j}\right);
$$

(g) follows since

$$
\begin{aligned}
&\mathcal{N}\left(\boldsymbol{x}_j^{o_i\cap m_j}\;\middle|\;\boldsymbol{x}_i^{o_i\cap m_j},\boldsymbol{\Sigma}_\kappa^{o_i\cap m_j}\right)\mathcal{N}\left(\boldsymbol{x}_j^{m_i\cap m_j}\;\middle|\;\boldsymbol{f},\mathbf{F}\right)\\
&\quad=\mathcal{N}\left(\left[\begin{array}{c}\boldsymbol{x}_j^{m_i\cap m_j}\\ \boldsymbol{x}_j^{o_i\cap m_j}\end{array}\right]\;\middle|\;\left[\begin{array}{c}\boldsymbol{f}\\ \boldsymbol{x}_i^{o_i\cap m_j}\end{array}\right],\right.\\
&\qquad\qquad\left.\left[\begin{array}{cc}\mathbf{F} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{\Sigma}_\kappa^{o_i\cap m_j}\end{array}\right]\right)\\
&\quad\equiv\mathcal{N}\left(\boldsymbol{x}_j^{m_j}\;\middle|\;\boldsymbol{a},\mathbf{A}\right);
\end{aligned}
$$

(h) follows because

$$
\begin{aligned}
p\left(\boldsymbol{x}_j^{m_j} \;\middle|\; \boldsymbol{x}_j^{o_j}\right) &= \sum_{\xi=1}^{Z} \pi_\xi^j \mathcal{N}\left(\boldsymbol{x}_j^{m_j} \;\middle|\; \boldsymbol{\mu}_\xi^{m_j}, \boldsymbol{\Sigma}_\xi^{m_j}\right) \\
&= \sum_{\xi=1}^{Z} \pi_\xi^j \mathcal{N}\left(\begin{bmatrix} \boldsymbol{x}_j^{m_i \cap m_j} \\ \boldsymbol{x}_j^{o_i \cap m_j} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu}_\xi^{m_i \cap m_j} \\ \boldsymbol{\mu}_\xi^{o_i \cap m_j} \end{bmatrix}, \right. \\
&\qquad \left. \begin{bmatrix} \boldsymbol{\Sigma}_\xi^{m_j[m_i m_i]} & \boldsymbol{\Sigma}_\xi^{m_j[m_i o_i]} \\ \left(\boldsymbol{\Sigma}_\xi^{m_j[m_i o_i]}\right)^T & \boldsymbol{\Sigma}_\xi^{m_j[o_i o_i]} \end{bmatrix}\right) \\
&\equiv \sum_{\xi=1}^{Z} \pi_\xi^j \mathcal{N}\left(\boldsymbol{x}_j^{m_j} \;\middle|\; \boldsymbol{b}, \mathbf{B}\right);
\end{aligned}
$$

(i) follows from being a product of Gaussians where

$$
\begin{aligned}
\mathbf{C} &= \left(\mathbf{A}^{-1} + \mathbf{B}^{-1}\right)^{-1} \\
\boldsymbol{c} &= \mathbf{C}\mathbf{A}^{-1}\boldsymbol{a} + \mathbf{C}\mathbf{B}^{-1}\boldsymbol{b} \\
z_c &= (2\pi)^{-d/2} |\mathbf{C}|^{+1/2} |\mathbf{A}|^{-1/2} |\mathbf{B}|^{-1/2} \\
&\times \exp\left\{-\frac{1}{2}\left[\boldsymbol{a}^T\mathbf{A}^{-1}\boldsymbol{a} + \boldsymbol{b}^T\mathbf{B}^{-1}\boldsymbol{b} - \boldsymbol{c}^T\mathbf{C}^{-1}\boldsymbol{c}\right]\right\};
\end{aligned}
$$

and (j) follows since the integral of a Gaussian is unity.

Thus, the Gaussian kernel between any two data points with incomplete data can be obtained analytically using (4). If a linear or polynomial kernel is chosen instead, the missing data can still be integrated out analytically. These cases are less interesting and quite trivial though. For instance, in the linear kernel case, analytically integrating out the missing data is equivalent to conditional mean imputation.

3. Experimental Results

We use a logistic regression classifier in this work. In logistic regression, the probability of label $y_i \in \{1, -1\}$ given the data point $\boldsymbol{x}_i$ is $p(y_i|\boldsymbol{x}_i) = \sigma(y_i \mathbf{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i))$, where $\sigma(z) = (1+\exp(-z))^{-1}$ is the sigmoid function. A data point $\boldsymbol{x}_i$ is embedded into feature space via the transformation

$$
\boldsymbol{\phi}(\boldsymbol{x}_i) = \begin{bmatrix} 1 & K(\boldsymbol{x}_i, \boldsymbol{x}_1) & \cdots & K(\boldsymbol{x}_i, \boldsymbol{x}_N) \end{bmatrix}
$$

where we use the positive semidefinite Gaussian kernel function K. As a result of this mapping, a nonlinear kernel classifier in the original input space can be constructed via a linear classifier in the transformed feature space. For a data set of N labeled data points, the (supervised) linear classifier $\boldsymbol{w}$ can be learned by maximizing the log-likelihood function $\ell(\boldsymbol{w}) = \sum_{i=1}^{N} \ln \sigma(y_i \mathbf{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i))$.

Our proposed kernel matrix completion method is driven by the GMM in (1). In (Ghahramani & Jordan, 1994), the algorithm is given for estimating a GMM from incomplete data via the expectation-maximization (EM) algorithm.

We compared our proposed method to two common imputation schemes on four data sets. The difference among the three methods is how the kernel matrix is computed. Our proposed method uses (4) to analytically integrate out the missing data for the kernel matrix. In conditional mean imputation, all missing data is "completed" with the conditional mean, which is obtained via the GMM in (2). Specifically, the missing features of each data point are replaced with their conditional mean:

$$
\boldsymbol{x}_i^{m_i} \leftarrow \mathbb{E}[\boldsymbol{x}_i^{m_i} | \boldsymbol{x}_i^{o_i}] = \sum_{\zeta=1}^{Z} \pi_\zeta^i \boldsymbol{\mu}_\zeta^{m_i}.
$$

In unconditional mean imputation, all missing data is "completed" with the unconditional mean, which does not require a model of the data. For example, if $\boldsymbol{x}_i$ is missing feature a (*i.e.,* $a \in m_i$), unconditional mean imputation will make the substitution

$$
x_i^a \leftarrow \mathbb{E}[x_i^a] = \frac{1}{M} \sum_{j=1}^{M} x_{s(j)}^a
$$

where there are M data points for which feature a was observed, and $s(j)$ is the index of the j-th such data point. The Gaussian kernel matrices for these two imputation methods were then computed as for a regular complete-data problem (using (3)).

Note that after obtaining the kernel matrix for each of the methods, we possess a standard complete-data classification problem to which any kernel-based algorithm can be applied. For each of the three methods, we used the same logistic regression classifier form. As a result, any differences in performance among the three methods are strictly the result of the kernel matrix calculation.

A measure of classifier performance is the area under a receiver operating characteristic curve (AUC), which is given by the Wilcoxon statistic (Hanley & McNeil, 1982)

$$
\text{AUC} = (MN)^{-1} \sum_{m=1}^{M} \sum_{n=1}^{N} \mathbf{1}_{x_m > y_n} \tag{5}
$$

where $x_1, \ldots, x_M$ are the classifier outputs of data belonging to class 1, $y_1, \ldots, y_N$ are the classifier outputs of data belonging to class -1, and $\mathbf{1}$ is an indicator function.

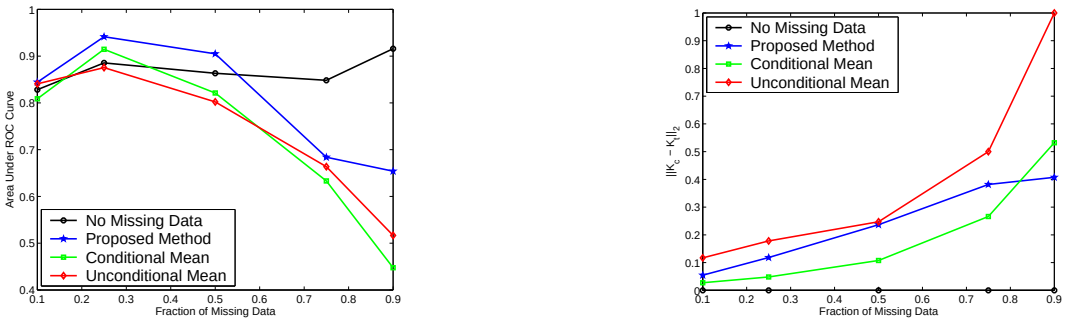


Figure 1. Experimental results on the IONOSPHERE data set. The figures show (a) AUC, and (b) the relative distances of the "estimated" kernel matrices from the "true" kernel matrix.

Table 1. Details of the 2-VIEW LAND MINE DETECTION data sets.

DATA SET	NUMBER OF MINES	NUMBER OF CLUTTER	NUMBER OF DATA POINTS WITH VIEW 1 ONLY	VIEW 2 ONLY	BOTH VIEWS	FRACTION OF MISSING DATA
AREA A	93	768	423	126	312	0.32
AREA B	139	693	134	146	552	0.17

3.1. Ionosphere

The proposed algorithm was first applied to the IONOSPHERE data set (from the UCI Machine Learning Repository), which has 351 data points and 34 features. In this example, the 34 features constitute 34 "views." Experimental results are shown in Figure 1(a) in terms of AUC, computed using (5). Each point on every curve is an average over 40 trials. Every trial consists of a random partition of training and testing data, and a random pattern of missing features (removed artificially). In every trial, 25% of the data was used as training data.

Since we artificially removed features, we can also build a classifier when there is no missing data. When no missing data exists, performance still varies as a function of the fraction of missing data because of the random partitions of training and testing data. From Figure 1(a), it can be seen that the proposed method consistently outperforms the imputation methods, with the most significant difference occurring when a large fraction of the data is missing. Remarkably, the proposed method sometimes achieves a larger AUC than that of the method with no missing data. We hypothesize that this phenomenon might occur if the missing feature values actually decrease or confuse class separation. In this case, their absence would be more beneficial than their presence.

We can also compute the Euclidean distance of each of the "estimated" kernel matrices (*i.e.,* the kernel matrices from the proposed or imputation methods) from the "true" kernel matrix (*i.e.,* the kernel matrix that would be obtained if all data was present). From Figure 1(b), it can be seen that as the fraction of missing data increases, the relative distance of the "estimated" kernel matrices to the "true" kernel matrix increases. Interestingly, the kernel matrix completed via conditional mean imputation is actually closer to the true kernel matrix than the proposed method's kernel matrix. We hypothesize that the proximity of the kernel matrix for the imputation method does not lead to better AUC because the single value imputation ignores the uncertainty of the missing data (Rässler, 2004).

3.2. 2-View Land Mine Detection

The proposed algorithm was also applied to two real data sets of 2-view land mine detection data. The goal for this data set is to classify mines (class 1) and clutter (class -1). The first view was an electro-optic infrared (EOIR) sensor, while the second view was a synthetic aperture radar (SAR) sensor. Data from each of the sensors were characterized by nine features. Details of these two data sets are summarized in Table 1.

For these experiments, 25% of the data was used as training data, while the remainder was used as testing data. Results shown in Table 2 are an average over 100 trials, where each trial represents a random partition of training and testing data. Since data is truly missing, no features are artificially removed.

Table 2. The mean AUC of 100 trials of each method for the 2-View Land Mine Detection data sets.

Data Set	Proposed Method	Conditional Mean Imputation	Unconditional Mean Imputation
Area A	0.6865	0.5604	0.6305
Area B	0.6579	0.5355	0.6171

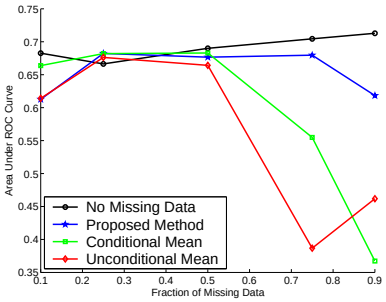


Figure 2. Experimental results in terms of AUC for the 4-View Land Mine Detection data set.

3.3. 4-View Land Mine Detection

The proposed algorithm was also applied to a real data set of 4-view land mine detection data. The goal for this data set is to again classify mines and clutter. The four views were a ground-penetrating radar (GPR) sensor, an EOIR sensor, a Ku-band SAR sensor, and an X-band SAR sensor. The sensors were characterized by 17, 6, 9, and 9 features, respectively. The data set had 713 total data points, only 91 of which were mines.

Unlike the 2-view data sets, every data point had data from each of the four views. Therefore, for the experiments, views (*i.e.,* sensors, or blocks of features) were randomly chosen to be artificially removed and thereafter treated as missing.

For the experiments, 25% of the data was used as training data, while the remainder was used as testing data. Each point in Figure 2 is an average over 10 trials, where each trial represents a random partition of training and testing data, and a random pattern of missing sensors (blocks of features).

From Figure 2, it can be seen that the proposed method again outperforms the imputation methods, with the most significant difference occurring when higher fractions of data are missing. The performance of the algorithm in which there is no missing data varies with the fraction of missing data because of the random partitions of training and testing data.

4. Conclusion

We have derived the expression for a Gaussian kernel function (or matrix) when faced with incomplete data. We analytically integrated out the missing data to obtain a closed-form expression for the kernel. As a result, incomplete data need no longer be a hindrance for general multi-view algorithms. We have demonstrated the superiority of this proposed method over two common imputation schemes, on both a benchmark data set as well as on three real multi-view land mine data sets. The advantage of the proposed method has been found to be most pronounced when a large amount of data is missing. The feature vectors for the multi-view land mine data sets will be made available to interested investigators upon request.

Analytical integration over the missing data can still be performed if one employs a linear or polynomial kernel instead of a Gaussian kernel, so our choice here of a Gaussian kernel is not overly restrictive. This kernel matrix completion work can also be utilized in semi-supervised algorithms. Many semi-supervised algorithms use the idea of a graph and the graph Laplacian (Zhu, Ghahramani & Lafferty, 2003), which can be directly computed from a kernel matrix. Future work will use our kernel matrix completion method to extend supervised algorithms to semi-supervised versions, when faced with incomplete data.

References

Ghahramani, Z. & Jordan, M. (1994). Supervised learning from incomplete data via the EM approach. In J. Cowan and G. Tesauro and J. Alspector (Eds.), *Advances in Neural Information Processing Systems 6*. San Mateo, CA: Morgan Kaufmann.

Graepel, T. (2002). Kernel matrix completion by semidefinite programming. *Proceedings of the International Conference on Artificial Neural Networks* (pp. 694–699).

Hanley, J. & McNeil, B. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology 143*, 29–36.

Lanckriet, G., Deng, M., Cristianini, N., Jordan, M., & Noble, W. (2004). Kernel-based data fusion and its application to protein function prediction in yeast. *Proceedings of the Pacific Symposium on Biocomputing 9* (pp. 300–311).

Rässler, S. (2004). *The impact of multiple imputation for DACSEIS* (DACSEIS Research Paper Series 5). University of Erlangen-Nürnberg, Nürnberg, Germany.

Schölkopf, B. & Smola, A. (2002). *Learning with kernels*. Cambridge: MIT Press.

Tsuda, K., Akaho, S., & Asai, K. (2003). The *em* algorithm for kernel matrix completion with auxiliary data. *Journal of Machine Learning Research 4*, 67–81.

Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using Gaussian fields and harmonic functions. *Proceedings of the Twentieth International Conference on Machine Learning*.