Proceedings of the
Workshop
on
Prior Knowledge for Text and Language Processing

Organizers:

Guillaume Bouchard
Hal Daumé III
Marc Dymetman
Yee Whye Teh

An ICML/UAI/COLT Workshop
Helsinki, Finland
9 July 2008

Organizers:

Guillaume Bouchard, *Xerox Research Centre Europe, France*

Hal Daumé III, *University of Utah, USA*

Marc Dymetman, *Xerox Research Centre Europe, France* (main contact)

Yee Whye Teh, *University College London, UK*

Programme Committee:

Guillaume Bouchard, *Xerox Research Centre Europe, France*

Nicola Cancedda, *Xerox Research Centre Europe, France*

Hal Daumé III, *University of Utah, USA*

Marc Dymetman, *Xerox Research Centre Europe, France*

Tom Griffiths, *University of California, Berkeley, USA*

Peter Grünwald, *Centrum voor Wiskunde en Informatica, Netherlands*

Kevin Knight, *University of Southern California, USA*

Mark Johnson, *Brown University, USA*

Yee Whye Teh, *University College London, UK*

Supported by:

PASCAL2
Pattern Analysis, Statistical Modelling and Computational Learning

Preface

The *Workshop on Prior Knowledge for Text and Language Processing* aims at presenting and discussing recent advances in machine learning approaches to text and natural language processing that capitalize on rich prior knowledge models in these domains.

Traditionally, in Machine Learning, a strong focus has been put on data-driven methods that assume little a priori knowledge on the part of the learning mechanism. Such techniques have proven quite effective not only for simple pattern recognition tasks, but also, more surprisingly, for such tasks as language modeling in speech recognition using basic n-gram models. However, when the structures to be learned become more complex, even large training sets become sparse relative to the task, and this sparsity can only be mitigated if some prior knowledge comes into play to constrain the space of fitted models. We currently see a strong emerging trend in the field of machine learning for text and language processing to incorporate such prior knowledge for instance in language modeling (e.g. through non-parametric Bayesian priors) or in document modeling (e.g. through hierarchical graphical models). There are complementary attempts in the field of statistical computational linguistics to build systems that do not rely uniquely on corpus data, but also exploit some form of *a priori* grammatical knowledge, bridging the gap between purely data-oriented approaches and the traditional purely rule-based approaches, that do not rely on automatic corpus training, but only indirectly on human observations about linguistic data. The domain of text and language processing thus appears as a very promising field for studying the interactions between prior knowledge and raw training data, and this workshop aims at providing a forum for discussing recent theoretical and practical advances in this area.

We would like to thank our authors, invited speakers, and panelists, as well as our Program Committee for helping make this workshop an interesting event. We would also like to thank the workshop chairs at ICML/UAI/COLT, Sanjoy Dasgupta and Michael Littman, as well as the Local Organization chair, Hannu Toivonen, for their support.

--- The Organizers: Guillaume Bouchard, Hal Daumé III, Marc Dymetman, Yee Whye Teh

This workshop is supported by the European Commission's PASCAL-2 Network of Excellence, and is part of its Thematic Programme "Leveraging Complex Prior Knowledge for Learning".

Workshop Program

Wednesday, July 9, 2008

9:00-9:05	Opening Remarks
9:05-9:50	Invited Talk: *Learning Rules: From PCFGs to Adaptor Grammars* Mark Johnson
9:50-10:00	Poster Preview
10:00-10:30	Coffee Break
10:30-10:55	*Constraints as Prior Knowledge* (pages 1-6) Ming-Wei Chang, Lev Ratinov and Dan Roth
10:55-11:40	Invited Talk: *Some thoughts on prior knowledge, deep architectures and NLP* Jason Weston
11:40-12:30	Poster Session
	Using Participant Role in Multiparty Meetings as Prior Knowledge for Nonparametric Topic Modeling (pages 21-24) Songfang Huang and Steve Renals
	Exponential family sparse coding with application to self-taught learning with text documents (pages 25-30) Honglak Lee, Rajat Raina, Alex Teichman and Andrew Y. Ng
	Using Prior Domain Knowledge to Build HMM-Based Semantic Tagger Trained on Completely Unannotated Data (pages 31-36) Kinfe Tadesse Mengistu, Mirko Hanneman, Tobias Baum and Andreas Wendemuth
	Knowledge as a Constraint on Uncertainty for Unsupervised Classification: A Study in Part-of-Speech Tagging (pages 37-42) Thomas J. Murray, Panayiotis G. Georgiou and Shrikanth S. Narayanan
	Dirichlet Process Mixture Models for Verb Clustering (pages 43-48) Andrea Vlachos, Zoubin Ghahramani and Anna Korhonen
12:30-14:30	Lunch

14:30-15:15	Invited Talk: *Incorporating Prior Knowledge into NLP with Markov Logic* Pedro Domingos
15:15-15:40	*Expanding a Gazetteer-Based Approach for Geo-Parsing Disease Alerts* (pages 11-14) Mikaela Keller, John S. Brownstein and Clark C. Freifeld
15:40-16:05	*Bayesian Modeling of Dependency Trees Using Hierarchical Pitman-Yor Priors* (pages 15-20) Hanna M. Wallach, Charles Sutton and Andrew McCallum
16:05-16:30	Coffee Break
16:30-17:30	Panel David Blei Fabrizio Costa Peter Grünwald Mark Johnson Jason Weston
17:30-17:55	*DRASO: Declaratively Regularized Alternating Structural Optimization* (pages 7-10) Partha Pratim Talukdar, Ted Sandler, Mark Dredze, Koby Crammer, John Blitzer and Fernando Pereira
17:55-18:00	Wrap-up

Constraints as Prior Knowledge

Ming-Wei Chang MCHANG21@UIUC.EDU
Lev Ratinov RATINOV2@UIUC.EDU
Dan Roth DANR@UIUC.EDU
Computer Science Department, University of Illinois at Urbana-Champaign

Abstract

Making complex decisions in real world problems often involves assigning values to sets of interdependent variables where an expressive dependency structure among these can influence, or even dictate, what assignments are possible. Commonly used models typically ignore expressive dependencies since the traditional way of incorporating non-local dependencies is inefficient and hence lead to expensive training and inference.

This paper presents Constrained Conditional Models (CCMs), a framework that augments probabilistic models with declarative constraints as a way to support decisions in an expressive output space while maintaining modularity and tractability of training. We develop, analyze and compare novel algorithms for training and inference with CCMs. Our main experimental study exhibits the advantage our framework provides when declarative constraints are used in the context of supervised and semi-supervised training of a probabilistic model.

1. Introduction

Decision making in domains such as natural language often involve assigning values to *sets* of interdependent variables where the expressive dependency structure among variables of interest can influence, or even dictate, what assignments are possible. To cope with these difficulties, problems are typically modeled as stochastic processes involving both output variables (whose values are sought) and information sources, often referred to as input or observed variables.

This work was supported by NSF grant NSF SoD-HCER-0613885, DARPA funding under the Bootstrap Learning Program and by MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC.

There exist several fundamentally different approaches to learning models that can assign values simultaneously to several interdependent variables (Punyakanok et al., 2005). Two extremes are to (1) completely ignore the output structure at the learning stage (by learning multiple independent models), while enforcing coherent assignments at the inference stage and (2) model, directly or indirectly, the dependencies among the output variables in the learning process and thus induce models that optimize a global performance measure. In the latter scenario, to allow efficient training and inference, assumptions on the probability distribution are made so that it is possible to factor the model into functions of subsets of the variables, yielding models such as Conditional Random Fields (CRFs) and Hidden Markov Models (HMMs).

However, in many problems, dependencies among output variables have non-local nature, and incorporating them into the model as if they were probabilistic phenomena can undo a great deal of the benefit gained by factorization, as well as making the model more difficult to design and understand. For example, consider an information extraction task where two particular types of entities cannot appear together in the same document. Modeling mutual exclusion in the scenario where n random variables can be assigned mutually exclusive values introduces n^2 pairwise edges in the graphical model, with obvious impact on training and inference. While efficient algorithms for leveraging a particular type of constraint can be developed, modeling of declarative non-local constraints this way is clearly very expensive. Moreover, a lot of parameters are being wasted in order to to learn something the model designer already knows.

This paper presents Constrained Conditional Models (CCMs). Generalizing and formalizing an approach introduced in (Roth & Yih, 2004; Roth & Yih, 2007), CCM is a framework that augments linear objective functions with declarative constraints as a way to support decisions in an expressive output space. CCMs

inject the constraints *directly* instead of doing it indirectly via a probability model. CCM allows the use of expressive constraints while keeping models simple and easy to understand. Factoring the models by separating declarative constraints naturally brings up interesting questions and calls for novel training and inference algorithms, as we discuss in this paper.

One interesting perspective is that the declarative constraints can be viewed as domain-specific knowledge which can be injected into the model in the supervised and, more interestingly, in the semi-supervised setting. We develop a formalism for constraints-based learning within the CCM framework. Our protocol can be used in the presence of any learning model, including those that acquire additional statistical constraints from observed data while learning. We experiment and report results with two models: maximum likelihood HMM (Rabiner & Juang, 1986) and its discriminative counterpart–the structured perceptron (Collins, 2002). We exhibit significant reduction in the number of training examples required in two information extraction problems. The results show that our approach yields very good results even in the presence of a small number of labeled examples.

2. Linear Models for Sequence Labeling Tasks

Although the discussion in this paper can be applied to other types of problems, we mainly focus on an important type of structured prediction problems: sequence labeling tasks. Given $\mathbf{x}$ as a series of tokens, we denote x_i as the i-th token of $\mathbf{x}$. Assuming there are T tokens in $\mathbf{x}$, the assignment $\mathbf{y}$ can be written as $y_1, y_2, \ldots y_T$, where y_i is the label for token x_i. The task in sequence labeling is to learn a model that can be used to predict the correct $\mathbf{y}$ given a new instance $\mathbf{x}$.

Linear models are the dominant family in machine learning, and can be represented as a weight vector $\mathbf{w}$, corresponding to a set of feature functions $\{\Phi\}$. For an input instance $\mathbf{x}$ and an output assignment $\mathbf{y}$, the "score" of the instance can be expressed as a weighted sum of feature functions: $f(\mathbf{x}, \mathbf{y}) = \sum w_i \phi_i(\mathbf{x}, \mathbf{y})$. Many different discriminative and generative learning algorithms can be represented as linear models. For example, models trained by Perceptron, naïve Bayes, and SVM, CRF and HMMs are linear models (Roth, 1999; Collins, 2002; Lafferty et al., 2001). Hidden Markov Model (HMM) is one of the most commonly used models for sequence labeling. Past works have shown that the prediction problem in HMMs can be viewed as a linear model over "local" features (Roth, 1999; Collins, 2002). That is, one can show that:

$$\operatorname*{argmax}_{\mathbf{y}} \log P(\mathbf{y}|\mathbf{x}) = \operatorname*{argmax}_{\mathbf{y}} \mathbf{w}^T \Phi(\mathbf{x}, \mathbf{y}), \qquad (1)$$

where $\mathbf{w}$ is a weight vector and Φ represents the feature functions, is an equivalent representation of HMM.

3. Training and Inference with Constraints

Although, in general, the feature functions $\Phi(\mathbf{x}, \mathbf{y})$ used in Eq. 1 can represent any function of $\mathbf{x}$ and $\mathbf{y}$, it is typical to encode local relationships only, (as in the linear representation of HMMs (Roth, 1999; Collins, 2002; Lafferty et al., 2001)) for tractable inference. However, such restriction usually renders the feature functions not expressive enough to capture non-local dependencies present in the problem.

In this paper, we propose the **Constrained Conditional Model (CCM)**, which provides a *direct* way to inject prior knowledge into a conditional model, in the form of **constraints**. The idea is that combining simple models with *expressive* constraints is a more effective approach to making probabilistic models expressive. Note that we do not increase the feature space explicitly by adding more conjunctive features but rather directly incorporate the constraints by augmenting the simple linear models. Since within CCMs we combine declarative constraints, possibly written as first order logic expressions (Rizzolo & Roth, 2007), with learned probabilistic models, we can treat CCMs as a way to combine or bridge logical expressions and learning statistical models.

Note that by modeling the constraints directly, the inference problem, Eq. 1, becomes harder to solve, compared to the one used by low order HMMs/CRFs. As we show later, such a sacrifice is usually very rewarding in terms of final performance; it is possible to use exact methods such as integer linear programming or approximate inference methods that we found to give good results.

3.1. Model

The formal definition of CCM is as follows.

We assume (1) a set of feature functions $\Phi = \{\phi_i(\cdot)\}$, $\phi_i : \mathcal{X} \times \mathcal{Y} \rightarrow R$, which typically encode *local* properties of a pair (x, y). (And often, the image of ϕ_i is $\{0, 1\}$); (2) a small set of constraints $C = \{C_i(\cdot)\}$, $C_i : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ that encode predicates over a pair (x, y); (3) a set of functions $d_{C_i} : \mathcal{X} \times \mathcal{Y} \rightarrow R$ that measure the degree to which the constraint C_i is violated in (x, y).

A **Constrained Conditional Model** can be repre-

sented using two weight vectors, $\mathbf{w}$ and ρ. The score of an assignment $\mathbf{y} \in \mathcal{Y}$ on an instance $x \in \mathcal{X}$ is obtained by:

$$f_{\Phi,C}(\mathbf{x},\mathbf{y}) = \sum w_i \phi_i(\mathbf{x},\mathbf{y}) - \sum \rho_i d_{C_i}(\mathbf{x},\mathbf{y}). \quad (2)$$

A CCM then selects as its prediction:

$$\mathbf{y}^* = \underset{\mathbf{y}}{\operatorname{argmax}}\, f_{\Phi,C}(\mathbf{x},\mathbf{y}). \quad (3)$$

Note that a CCM is not restricted to be trained with any particular learning algorithm. Similar to other linear models, specialized algorithms may need to be developed to train CCMs. Unlike standard linear models, we assume the availability of some prior knowledge, encoded in the form of *constraints*, when learning a CCM. When there is no prior knowledge, there is no difference between CCMs and linear models.

Although the two terms of Eq. 2 may appear similar, they are very different in several aspects. Essentially, a predicate $C(\mathbf{x},\mathbf{y})$ is viewed as a "first order logical expression", which is very different from features $\Phi(\mathbf{x},\mathbf{y})$. Due to their first order logic nature, the set of constraints is compact. (In our experiments, we only have about 10 constraints, compared to thousands of features in a feature vector). Moreover, $C(\mathbf{x},\mathbf{y})$ usually encodes long distance relationships among $\mathbf{y}$ variables, which cannot be captured by the feature functions $\Phi(\mathbf{x},\mathbf{y})$. For example, $C(\mathbf{x},\mathbf{y})$ might be "1, if all y_is in the sequence y are assigned different values, 0 otherwise", which is difficult to model using features.

Importantly, we separate the constraints from features in Eq. 2 because we know that the constraints should be trusted most of the time. Therefore, the penalties ρ can be fixed or handled **separately**. If we are confident about our knowledge, rather than learning the $\{\rho_i\}$, we can directly set them to ∞, thus enforcing the chosen assignment $\mathbf{y}$ to satisfy the constraints. It is important to note that although ρ_i is fixed, it may still impact the learning of the weights w_i (this point will be explained in detail in Section 3.3).

3.2. Inference with Constraints

In the earlier related works that made use of constraints, the constraints were assumed to be binary functions; in most cases, a high level (first order logic) description of the constraints was compiled into a set of linear inequalities, and exact inference was done using a integer linear programming formulation(ILP) (Roth & Yih, 2004; Roth & Yih, 2007; Barzilay & Lapata, 2006; Clarke & Lapata, 2006). Intractable in principle, ILP proved to be quite successful in practice, since the constraints were very sparse (a small number of $\mathbf{y}$ variables present in each constraint) (Roth & Yih, 2007).

However, in our CCM formalism, rather than using binary constraints, we introduce a "degree of violation" to each constraint. The significance of this is that it is possible that a label assignment violates the constraints in more than one place. Therefore, if binary function is used, once the value is set to 1, the algorithm loses the ability to discriminate constraint violations in other locations of the same instance. Note that even with such a choice, ILP can still be applied to solve the inference problem Eq. 3. However, here we choose not to do it, but rather to approximate the degree of violation incrementally, by estimating it over an incomplete label assignment. This allows us to design a search procedure which finds an approximate solution to Eq. 3 efficiently. In this work, we rewrite the constraint function as:

$$d_{C_i}(\mathbf{x},\mathbf{y}) = \sum_{t=1}^{T} \hat{C}_i(\mathbf{x}; y_1, \ldots, y_t),$$

where T is number of tokens in this instance, and $\hat{C}_i(\mathbf{x}; y_1, \ldots, y_t)$ is a binary function which approximates the predicate C_i, by computing it over the t-prefix of the assignment $\mathbf{y}$, $(\mathbf{x}; y_1, \ldots, y_{t-1})$.

We use this estimation to guide the search procedure for optimizing the objective function in Eq. 3 with partially labeled sequence. In this paper, we use beam search as our search procedure. A* search can be also applied here with admissible heuristic if the ρ_is are positive for all constraints. Note that this approximation methods may not work for all types of constraints. For example, constraints such as "label A must appear at least once in the sequence", do not have "degree" of violation. For these constraints, the function d_C is the identity function, essentially making them binary constraints; these constraints are examined only at the end of the search procedure.

3.3. Training with CCM

In this section, we propose and describe several approaches of training CCMs. There are two independent decisions to be made, leading to four different training strategies.

The first decision is whether we want to use *factored* approaches or *joint* approaches. *Factored* approaches treat the first term (feature term) and the second term (constraints term) of Eq. 2 separately. That is, $\mathbf{w}$ and ρ are learned *independently*. This approach is also referred to *Learning Plus Inference* (L+I) (Punyakanok et al., 2005), since we learn the models separately but

put the constraints back into consideration at testing time. The *joint* approach, which we call *Inference Based Training* (IBT) (Punyakanok et al., 2005), learns $\mathbf{w}$ and ρ together during training by using the true objective function with both terms in Eq. 3.

The second decision is whether we want to use hard constraints or weighted constraints. Using hard constraints is equivalent to setting ρ to ∞; in this case, the notion of "degree" no longer exists, the constraints essentially become Boolean functions, and we do not output assignments which violate them. Using weighted constraints is important if we know that the prior knowledge does not hold all the time and it also means that we need to figure out the penalty ρ for each constraint from labeled data.

Training CCMs with factored approaches is simple, since factored approaches learn $\mathbf{w}$ and ρ independently. $\mathbf{w}$ can be learned with standard algorithms for training linear models. If we chose to use hard constraints, the training procedure is complete, given that the penalty of each constraint is infinity. In this paper, this approach is called **L+CI** (Learning Plus Constrained Inference) . However, it is often the case that the prior knowledge is not perfect, or that the weights for every constraint should be different. To figure out the penalty for each constraint, in this case, we count how many times it is violated in the labeled data, and reduce the penalty coefficients for those violated constraints (refer to (Chang et al., 2008) for details). This approach is called **L+wCI** (Learning Plus weighted Constrained Inference).

Alternatively, we can enforce the constraints during training as well as testing. In this approach, *Inference Based Training* (IBT), the constraints may be hard or soft, resulting in **CIBT** and in **wCIBT** respectively. Our IBT training algorithms are based on the Perceptron update rule.

The pseudocode for **CIBT** and **wCIBT** is given in Algorithm 1, which is similar to the perceptron algorithm. However, the constraints are taken into account during the training procedure. CIBT is a more conservative update rule than L+CI, since when the constraints term "corrects" the label assignment, no update will be performed. Note that when weighted constraints are used, the algorithm also updates the penalty ρ during the training procedure.

Since the constraints in Eq. 2 have non-local nature, we give up exact inference (with dynamic programming) and use beam search to find an approximate solution. The idea of using non-local features in perceptron was also explored in (Collins & Roark, 2004) that used beam search for inference with application to syntactic parsing. Later, (H. Daumé & Marcu, 2005) extended this idea to other applications. While wCIBT uses a similar algorithm to assign weights to the constraints, it differs from (Collins & Roark, 2004; H. Daumé & Marcu, 2005) in the nature of the "features": there, a large number of weights are assigned to "propositional" non-local features in perceptron, while we assign a small number of weights to constraints that are high level, 'first order logic' predicates.

Algorithm 1 IBT training: CIBT & wCIBT

Require: D is the training dataset, K is the number of constraints, M is the number of iterations

for $i = 1 \dots K$ **do**
  $if(hardConstraints)$ then $\rho_i = \infty$ else $\rho_i = 0$
end for
for $i = 1 \dots M$ **do**
  for $(\mathbf{x}, \mathbf{y}^*) \in D$ **do**
    $\hat{\mathbf{y}} = \text{argmax}_{\mathbf{y}}[\sum w_i \phi_i(\mathbf{x}, \mathbf{y}) - \sum \rho_i d_{C_i}(\mathbf{x}, \mathbf{y})]$
    $\mathbf{w} = \mathbf{w} + \Phi(\mathbf{x}, \mathbf{y}^*) - \Phi(\mathbf{x}, \hat{\mathbf{y}})$
    if weightedConstraints **then**
      $\rho = \rho + d_C(\mathbf{x}, \mathbf{y}^*) - d_C(\mathbf{x}, \hat{\mathbf{y}})$
    end if
  end for
end for

3.4. Semi-Supervised Learning with CCM

Acquiring labeled data is a difficult and expensive task. Therefore, an increasing attention has been recently given to semi-supervised learning, where large amounts of unlabeled data are used to improve models learned from a small training set (Haghighi & Klein, 2006; Thelen & Riloff, 2002). In this section, we present COnstraint-Driven Learning (**CODL**), an algorithm that uses constraints as prior knowledge in semi-supervised setting (Chang et al., 2007) and show that prior knowledge plays a crucial role when the amount of labeled data is limited. CODL makes use of CCM, which provides a good platform to combine the learned models and prior knowledge.

As is often the case in semi-supervised learning, the algorithm can be viewed as a process that improves the model by generating feedback through *labeling* unlabeled examples. *CODL* pushes this intuition further, in that the use of constraints allows us to better exploit domain information as a way to label unlabeled examples, along with the current learned model. Given a small amount of labeled data and a large unlabeled pool, *CODL* initializes the model with the labeled data and then repeatedly: **(1)** uses the learned model and the *constraints* to label the unlabeled instances,

and **(2)** updates the model via the newly labeled data.

Algorithm 2 COnstraint **D**riven **L**earning (CODL): Using constraints to guide semi-supervised learning.

Require: labeled training set $\mathbf{L}$; unlabeled dataset $\mathbf{U}$; N learning cycles; a balancing parameter with the supervised model γ; a set of constraints $\{C\}$; a supervised learning algorithm $learn(.)$

1: Init: $(\mathbf{w}, \rho) = (\mathbf{w_0}, \rho_\mathbf{0}) = \mathbf{learn}_{[(\mathbf{w})\mathbf{CIBT}/\mathbf{L+(w)CI}]}(\mathbf{L})$.
2: **for** N iterations **do**
3: $\quad\mathbf{T} = \emptyset$
4: $\quad$**for** $\mathbf{x} \in \mathbf{U}$ **do**
5: $\quad\quad(\mathbf{x}, \hat{\mathbf{y}}) = \mathbf{InferenceWithConstraints}(\mathbf{x}, \mathbf{w}, \rho, \{C_i\})$
6: $\quad\quad\mathbf{T} = \mathbf{T} \cup \{(\mathbf{x}, \hat{\mathbf{y}})\}$
7: $\quad$**end for**
8: $\quad(\mathbf{w}, \rho) = (1 - \gamma)\mathbf{learn}_{[(\mathbf{w})\mathbf{CIBT}/\mathbf{L+(w)CI}]}(\mathbf{T}) + \gamma(\mathbf{w_0}, \rho_\mathbf{0})$
9: **end for**

CODL is summarized in Algorithm 2. *CODL* initializes the model with traditional supervised learning on a small labeled set $\mathbf{L}$ (line 1). The supervised learning algorithm $learn_{[(w)CIBT/L+(w)CI]}(.)$ used in lines 1 and 8, learns $(\mathbf{w}, \rho)$ jointly if the wCIBT approach is used. If the L+wCI approach is used, it learns $\mathbf{w}$ independently from estimating ρ. If CIBT or L+CI is used, the learning algorithm $learn_{[(w)CIBT/L+(w)CI]}(.)$ always sets ρ to infinity.

Line 8 in the algorithm should be further clarified. (Nigam et al., 2000) shows that semi-supervised training can degrade the learned model's performance and suggests to balance the contribution of labeled and unlabeled data. The parameter re-estimation in line 8 uses a similar intuition, but instead of weighting data instances, we introduce a smoothing parameter γ which controls the convex combination of models induced by the labeled and unlabeled data. Unlike the technique mentioned above, which focuses on naïve Bayes, our method allows us to weight linear models generated by different learning algorithms. Due to space limitations we do not address several other important issues related to the algorithm, for more details, please refer to (Chang et al., 2008).

4. Experiments and Results.

We applied our approach to two information extraction tasks: extracting fields from citations and advertisements. Since in both problems, the fields are typically related and interdependent, these kinds of applications provide a good test case for an approach like ours (the data for both problems is available at: `http://L2R.cs.uiuc.edu/~cogcomp/Data/IE.tgz`.). Due to space restrictions, we omit the details of the datasets, and report only the main results, omitting the analysis constraints' utility, sensitivity to constraint violation penalty, etc. The reader is referred to (Chang et al., 2008) for additional details.

Citations	
Start	The citation must start with author or editor.
AppearsOnce	Each field must be a consecutive list of words, and can appear at most once in a citation.
Punctuation	State transitions must occur on punctuation marks.
BookJournal	The words *proc, journal, proceedings, ACM* are *JOURNAL* or *BOOKTITLE*.
...	...
TechReport	The words *tech, technical* are *TECH_REPORT*.
Title	Quotations can appear only in titles.
Location	The words *CA, Australia, NY* are *LOCATION*.

Table 1. The list of constraints used in the citations domain. Some constraints are relatively difficult to represents in traditional models.

Table 1 illustrates the list of constraints for the citations domain. We measured token-level accuracy of the learned models and evaluated the impact of the constraints in the supervised and semi-supervised settings. Table 2 shows the results for HMM (trained in a maximum-likelihood way). The results highlight the effect of applying the constraints. A semi-supervised model driven by constraints and 20 labeled samples, using L+wCI, is competitive with the traditional HMM trained with 300 labeled samples.

Table 3 compares the discriminative approaches for structured perceptron (the baseline, without constraints, is denoted $\mathbf{L}$). It can be seen that while CIBT seems like a reasonable strategy, it does not perform well. L+CI performs better than the baseline structured perceptron and CIBT. Moreover, consistently with (Punyakanok et al., 2005), for a small number of examples, L+CI outperforms all other algorithms while, when the amount of training data is large enough, learning the constraint violation penalties from the data (wCIBT) achieves the best results.

As observed already in the literature (see for example (Ng & Jordan, 2001)), with small amounts of labeled data, maximum-likelihood (ML) training approaches outperform discriminative ones. However, for sufficient amounts of data, and without constraints, the discriminative approach outperforms the ML approach. With 300 training samples on the citations domain, the structured perceptron achieves accuracy of 89.83% on the citations domain versus 86.35%, achieved by ML HMM when trained on the same

amount of labeled data. However, when learning constraint violation penalties, the ML approach consistently outperformed the discriminative approach. One reason for that is that in L+wCI in ML approach, we assume that the constraints hold by default, and reduce the constraint violation penalty only when the labeled data violates the constraints. On the other hand, in the wCIBT approach in discriminative setting, we learn constraint violation penalties from scratch. More data must be needed for successful training. Moreover, despite trying several learning strategies, we could not achieve improvements with the semi-supervised training for the discriminative approach.

Citations(Maximum Likelihood HMM)				
	Supervised		Semi-Supervised	
#Train	HMM	L+wCI	HMM	L+wCI
5	58.48	70.85	64.39	**77.09**
10	68.61	75.11	70.34	**81.25**
20	70.81	81.31	75.83	**85.00**
300	86.66	94.08	87.80	**94.51**

Table 2. The impact of using constraints for supervised and semi-supervised learning (generative HMM) with 5,10,20,300 labeled training samples.

5. Conclusions

This paper provides a unified view of a framework aimed to facilitate decision making with respect to multiple interdependent variables the values of which are determined by learned probabilistic models. We proposed CCM, a framework that augments linear models with expressive declarative constraints as a way to support decisions in an expressive output space while maintaining modularity and tractability of training. Importantly, this framework provides a principled way to incorporate expressive background knowledge into the decision process. It also provides a way to combine conditional models, learned independently in different situations, along with declarative information to support coherent global decisions.

	Supervised setting. Structured Perceptron-Citations Domain			
#Train	L	L+CI	CIBT	wCIBT
5	50.14	**66.36**	64.79	61.65
10	59.90	**72.91**	68.52	69.64
20	68.26	77.28	72.79	**78.46**
300	89.83	91.63	87.83	**93.89**

Table 3. Comparison between discriminative learning strategies. L+CI outperforms L while $CIBT$ performs poorly. $wCIBT$ achieves the best results when enough data is used.

References

Barzilay, R., & Lapata, M. (2006). Aggregation via Set Partitioning for Natural Language Generation. *Proc.of HLT/NAACL.*

Chang, M., Ratinov, L., & Roth, D. (2007). Guiding semi-supervision with constraint-driven learning. *Proc. of the ACL.*

Chang, M., Ratinov, L., & Roth, D. (2008). Structured learning with constrained conditional models. *In Submission.*

Clarke, J., & Lapata, M. (2006). Models for sentence compression: A comparison across domains, training requirements and evaluation measures. *Proc. of ACL.*

Collins, M. (2002). Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. *Proc. of EMNLP.*

Collins, M., & Roark, B. (2004). Incremental parsing with the perceptron algorithm. *Proc. of the ACL.*

H. Daumé, I., & Marcu, D. (2005). Learning as search optimization: approximate large margin methods for structured prediction. *Proc. of ICML.*

Haghighi, A., & Klein, D. (2006). Prototype-driven learning for sequence models. *Proc. of HTL-NAACL.*

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proc. of ICML.*

Ng, A. Y., & Jordan, M. I. (2001). On discriminative vs. generative classifiers: A comparison of logistic regression and naïve bayes. *Proc. of NIPS* (pp. 841–848).

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

Punyakanok, V., Roth, D., Yih, W., & Zimak, D. (2005). Learning and inference over constrained output. *Proc.of IJCAI.*

Rabiner, L. R., & Juang, B. H. (1986). An introduction to hidden markov models. *IEEE ASSP Magazine*, 4–16.

Rizzolo, N., & Roth, D. (2007). Modeling Discriminative Global Inference. *Proc. of the ICSC* (pp. 597–604). IEEE.

Roth, D. (1999). Learning in natural language. *Proc.of IJCAI.*

Roth, D., & Yih, W. (2004). A linear programming formulation for global inference in natural language tasks. *Proc. of CoNLL.*

Roth, D., & Yih, W. (2007). Global inference for entity and relation identification via a linear programming formulation. *Introduction to Statistical Relational Learning.* MIT Press.

Thelen, M., & Riloff, E. (2002). A bootstrapping method for learning semantic lexicons using extraction pattern contexts. *Proc. of EMNLP.*

Expanding a Gazetteer-Based Approach for Geo-Parsing Disease Alerts

Mikaela Keller MIKAELA.KELLER@CHILDRENS.HARVARD.EDU
John S. Brownstein JOHN.BROWNSTEIN@CHILDRENS.HARVARD.EDU
Clark C. Freifeld CLARK.FREIFELD@CHILDRENS.HARVARD.EDU
Children's Hospital Informatics Program at the Harvard-MIT Division of Health Sciences and Technology, 300 Longwood Ave., Boston, MA 02115 USA

Abstract

Discovering in a text the geographic references it may contain, is a task that human readers perform using both their lexical and contextual knowledge. Using a gazetteer to label such targeted references in a dataset, this paper proposes an approach to learning the context in which they appear and by this means extending the prior knowledge encoded in the gazetteer. The present work was carried in the particular framework of a system for disease outbreak alerts detection and geo-indexing.

1. Introduction

When presented in a text, with a phrase that is out of his vocabulary, a human reader would most likely be able to guess whether this phrase refers to a geographic location or not. This reader would infer the semantic role of the phrase with a certain accuracy, because he has a prior knowledge on the syntactic context on which geographic references appear, maybe also on their particular character distribution or on the fact that they generally begin with a capital letter, etc. There have been a number of approaches, exploiting this kind of prior knowledge to named entity recognition and more generally to information extraction problems (see *eg* (Tjong Kim Sang & De Meulder, 2003; Carreras & Màrquez, 2005)). They often rely on complex feature sets to represent the words and on heavily annotated datasets to account for the human experience.

HealthMap (Brownstein & Freifeld, 2007; Freifeld et al., 2008) is a system that automatically monitors disease outbreak alerts in news media from all around the world. It queries news aggregators such as Google News, but also news sources curated by experts, for relevant reports. It filters the retrieved documents into several taxonomies and provides on its website, `www.HealthMap.org`, a geographic and by-disease display of the ongoing alerts. Its operating mechanism can be naturally decomposed in a number of easily identifiable Information Retrieval and Natural Language Processing tasks, such as document retrieval, document categorization, information extraction, etc. In the present work we are interested in a sub-task of the last phase of the information processing scheme: the geographic parsing ("geo-parsing") (Woodruff & Plaunt, 1994) of a disease outbreak alert or the extraction from one such textual document of the related geographic information needed for the precise mapping into a world map.

So far, HealthMap assigns incoming alerts to a low resolution geographic description such as its country, and in some cases its immediately lower geographic designation (for the USA and Canada, it would provide for example the state). The system uses a rule-based approach relying on a purposely crafted gazetteer, which was built incrementally by adding relevant geographic phrases extracted from the specific kind of news report intended for mapping. The approach consists roughly in a look-up tree algorithm which tries to find a match between the sequences of words in the alert and the sequences of words in the entries of the gazetteer. It also implements a set of rules which use the position of the phrase in the alert to decide whether or not the phrase is related to the reported disease.

The gazetteer contains around 4000 key phrases, some of which refers to geographic locations with several resolution levels (from hospitals' to countries'), some are negation phrases (≈ 500 phrases, *eg Brazil nut* or *turkey flock* are not considered location references) as well as phrases that are specific to the kind of data processed (*Center for Disease Control*, *Swedish health officials*, etc.).

HealthMap is interested in developing a higher resolution in the geographic assignments outside of those contained in the gazetteer. The question we would like to answer is whether we can use the prior knowledge encoded in the gazetteer to expand the system capabilities in the geographic parsing of the alerts.

Appearing in *Proceedings of the 25^{th} International Conference on Machine Learning*, Helsinki, Finland, 2008.

2. Our Approach

The basic idea behind our approach is to have a dataset of alerts tagged with the gazetteer-based algorithm as well as with more general linguistic knowledge (*eg* part-of-speech tags, etc.), and then to use this dataset with tags partially hidden to learn a generalization of the parsing. In the toy example of Fig. 1, a sentence is enhanced with its corresponding part-of-speech tags and gazetteer-based geo-parsing tags (the blue rectangle). In order to learn a generalization of the geo-parsing, the same sentence would be used in our training dataset with the specific identity of the word *New Caledonia*, hidden, but its part-of-speech, preserved.

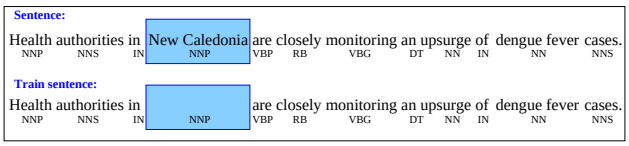

Figure 1. Example of training data.

Our dataset consists of disease outbreak alerts retrieved in 2007 by the HealthMap system. We tagged them with the part-of-speech tagger provided by NEC's project SENNA (Collobert & Weston, 2007), which has a reported accuracy of 96.85%. Provided that, in English, location names often begin with capital letters or appear as acronyms, we assigned to the words in the alerts, in addition to their part-of-speech tags, a capitalization status, *ie* none, first character, upper case. We used the rule-based approach to tag the words in the alert that match geographical references found in the gazetteer. Since the alerts in the dataset have been displayed (with the supervision of an expert) in the HealthMap world map, we were able in addition, to distinguish among the assigned tags, the ones that actually referred to a location where the event was taking place (location IN or *locIN*), from those that were foreign mentions (location OUT or *locOUT*).

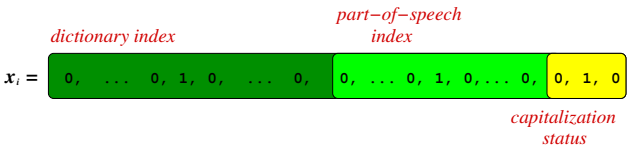

Figure 2. Words sparse representation.

From a machine learning perspective, our dataset is composed of alerts examples $\mathbf{x} = [x_1, \ldots, x_L]$ of length L, and their corresponding location labels $\mathbf{y} = [y_1, \ldots, y_L]$, $y_i \in \{None, locIN, locOUT\}$. The words x_i are represented by their part-of-speech tag, their capitalization status and occasionally by their index in the dictionary $\mathcal{D}$, extracted from the training dataset. Figure 2 illustrates the vectorial representation of words. The $|\mathcal{D}|$ (size of $\mathcal{D}$) first components of x_i correspond to the dictionary indexes and are all equal to zero, except for the position coinciding with the word index in $\mathcal{D}$. Similarly, the next K features of x_i correspond to the part-of-speech tag indexes in the K part-of-speech tag list. And finally, the last three features stand for the three possible capitalization status. As explained previously, one important characteristic of this experiment, is the fact that words are only partially accessible to the learning algorithm. We applied a lower bound $minfreq$ on the word frequency in the dataset to decide whether or not a word index was to be hidden. Only frequent word indexes are visible to the model. This was implemented as a dictionary cut-off, in which infrequent words are removed from the dictionary. Table 1 reports the resulting sub-dictionaries size for varying $minfreq$, in two training datasets T_0 and T_1 which respectively have 1000 and 2500 alerts. An out-of-dictionary word will have its $|\mathcal{D}|$ first components equal to zero.

$minfreq$	0	4	10	20
T_0 dict. size	15,000	5,000	3,000	1,700
T_1 dict. size	25,300	9,500	5,900	3,900

Table 1. Sizes of (sub-)dictionaries extracted from the training datasets T_0 (1000 alerts) and T_1 (2500 alerts).

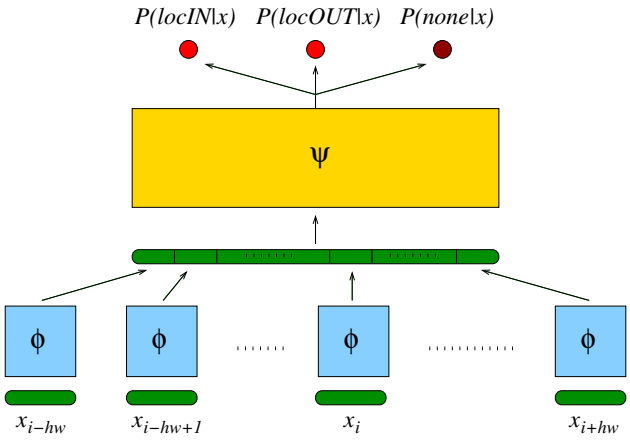

Figure 3. Illustration of the neural network.

We trained a neural network, by negative log-likelihood minimization to output a probability estimate of the label y_i value corresponding to the word x_i in i^{th} position of an alert $\mathbf{x}$,

$$NN(i, \mathbf{x}) = P(y_i | x_{i-hw}, \ldots, x_i, \ldots, x_{i+hw})$$

given a window $(n - 1 = 2 \times hw)$ of preceding and following words.

$minfreq$	0	4	10	20
% w/o index (T_0 dict.)	4.9	9.7	13.6	18.1
% loc w/o index (T_0 dict.)	16.8	36.6	47.3	61.4
% w/o index (T_1 dict.)	3.0	5.8	8.1	10.6
% loc w/o index (T_1 dict.)	6.6	22.1	31.3	38.7

Table 2. Percentage among the validation set words and words labeled as locations, of words without index in the dictionary.

The neural network, illustrated in Figure 3, can be decomposed as follow. First, each word in the window sequence is given in input to the same multi-layer perceptron (MLP) which has been replicated $n = 2 \times hw + 1$ times, in a siamese network fashion (Bromley et al., 1993). This first MLP can be seen as a function ϕ mapping the extremely sparse representation x_i of the words into a new representation, $\phi(x_i) \in \mathbf{R}^d$, which has the advantage of being learned during the training. This approach was applied with success for language modelling in (Bengio et al., 2003) and more recently for semantic role parsing in (Collobert & Weston, 2007). The outputs $\phi(x_{i-hw}), \ldots, \phi(x_i), \ldots, \phi(x_{i+hw})$ are concatenated into a vector $z \in \mathbf{R}^{d \times n}$ which is itself given in input to a second multi-layer perceptron. This second MLP, called ψ in Figure 3, has as output layer a *softmax* filtering function which allows us to consider the outputs of the neural network as probabilities.

3. Results

We trained several such neural networks on the two datasets T_0 (1,000 alerts) and T_1 (2,500 alerts), with extracted dictionaries of varying sizes according to our lower bound $minfreq$, as described in Table 1. We tested the models obtained on a separated validation set of 1000 alerts (465,297 words to tag, 6,156 with target *locIN* and 5,013 with *locOUT*). Table 2 summarizes the percentage in that set, of words that, as a consequence of the dictionaries cut-off, do not have an index in the dictionaries extracted from T_0 and T_1. The 2nd and 4th lines of Table 2 show the out-of-dictionary percents among the words that were assigned a location tag. The much higher proportions confirms the intuition that individual location mentions are infrequent words. Note that in the first column of Table 2, where no word is removed from the original dictionaries, there is still a certain amount of out-of-dictionary words. This is due to the fact that the vocabulary of the validation dataset is not completely covered by the dictionaries extracted from the training datasets.

Given the approximate nature of the solution found when training neural networks by stochastic gradient descent we repeated the learning process for each condition 5 times to estimate the variance. Figure 4 displays the results in terms of the obtained F_1 score for the parsing of the words targeted with tags location IN and location OUT, as well as the F_1 score obtained if we do not make the distinction between the two of them (referred as *loc*), with models trained both on T_0 and T_1. There is a discrepancy in the results for the tag *locIN / locOUT* and the reported *loc* tag, showing that the neural networks confuse *locINs* for *locOUTs* targets, and *vice versa*. That seems to suggest that our set of features is not suited to fully make this distinction. The size of the window in particular which we kept relatively small ($hw = 4$) for these preliminary experiments, may help to narrow this gap. If we consider the performance on the whole location tags, however, the results seem to be encouraging. Another encouraging facts is that increasing the size of the training dataset improves the performance of the models.

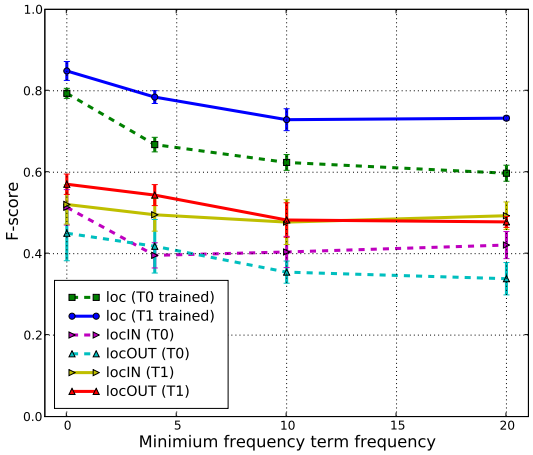

Figure 4. F1-score of *locIN*, *locOUT*, and the grouped location labels for two sizes of training dataset (train0: 1000 alerts, train1: 2500 alerts)

In Figure 5, to emphasize the fact that this approach do differ from the gazetteer-based approach, the results obtained with models trained on T_1 have been sliced into the F_1 scores of words that were represented with and without their dictionary index feature. Results for the words that were not targeted as locations (None tags), are also reported in Figure 5, they oscillated around an F_1 score of 99% for words with their index feature and 95% for those without.

The observed increase in performance for no-index words proportionally to the size of the dictionary cut-off suggest a potential for discovering phrases out of the initial gazetteer, and that, without having a too high lost in performance for unhidden words. A visual inspection of the false positive among the words without index, reveals that indeed many among the words labeled as *None* that the system decided

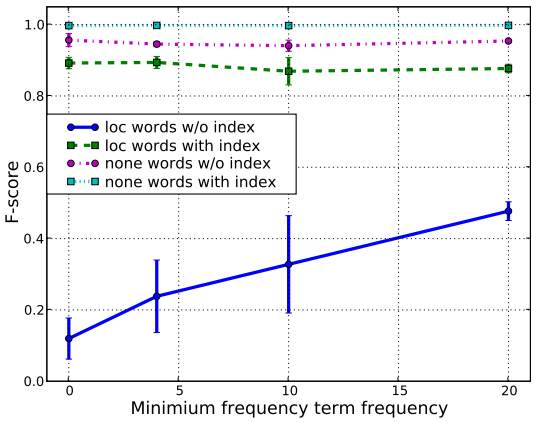

Figure 5. F1-score of *loc* and *none* targets for words with and without dictionary index feature in their representation.

were location are in fact location references that are out of the gazetteer.

4. Conclusion

We have presented a promising approach to incorporate the prior knowledge encoded in a rule-based procedure into a more general statistical framework. We have demonstrated that the described model has the ability to discover geographic references based solely on the context they appear in. The experiments also attested that providing additional training material improves the performance of the model, suggesting that despite the fabricated nature of the data it is still able to dispense interesting information. We plan in the close future to try more complex features for the word representations such as *eg* their semantic role labels. This technique could be integrated to a more conventional method for geo-parsing based on a geographically annotated dataset. It would be interesting to evaluate the contribution this approach could provide to the final task of indexing disease outbreak reports for geographic information retrieval.

References

Bengio, Y., Ducharme, R., Vincent, P., & Gauvin, C. (2003). A Neural Probabilistic Language Model. *JMLR*, *3*, 1137–1155.

Bromley, J., Guyon, I., LeCun, Y., Sackinger, E., & Shah, R. (1993). Signature Verification using a Siamese Time Delay Neural Network. *Advances in Neural Information Processing Systems 6*.

Brownstein, J. S., & Freifeld, C. C. (2007). Healthmap: the development of automated real-time internet surveillance for epidemic intelligence. *Euro Surveill*, *12(48)*, 3322.

Carreras, X., & Màrquez, L. (2005). Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. *Proceedings of the 9th Conference on Natural Language Learning, CoNLL-2005*. Ann Arbor, MI USA.

Collobert, R., & Weston, J. (2007). Fast semantic extraction using a novel neural network architecture. *Proceedings of 45th Annual Meeting of the Association for Computational Linguistics*.

Freifeld, C. C., Mandl, K. D., Reis, B. Y., & Brownstein, J. S. (2008). Healthmap: Global infectious disease monitoring through automated classification and visualization of internet media reports. *J Am Med Inform Assoc*, *15(2)*, 150–157.

Tjong Kim Sang, E. F., & De Meulder, F. (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL-03)* (pp. 142–147). Edmonton, Canada.

Woodruff, A. G., & Plaunt, C. (1994). Gipsy: Automated geographic indexing of text documents. *Journal of the American Society for Information Science*, *45:9*, 645–655.

DRASO: Declaratively Regularized Alternating Structural Optimization

Partha Pratim Talukdar PARTHA@CIS.UPENN.EDU
Ted Sandler TSANDLER@CIS.UPENN.EDU
Mark Dredze MDREDZE@CIS.UPENN.EDU
Koby Crammer CRAMMER@CIS.UPENN.EDU
Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104 USA

John Blitzer JOHN@BLITZER.COM
Microsoft Research Asia, Hai Dian District, Beijing, China 100080

Fernando Pereira PEREIRA@GOOGLE.COM
Google, Inc., 1600 Amphitheatre Parkway, Mountain View, CA 94043 USA

Abstract

Recent work has shown that Alternating Structural Optimization (ASO) can improve supervised learners by learning feature representations from unlabeled data. However, there is no natural way to include prior knowledge about features into this framework. In this paper, we present Declaratively Regularized Alternating Structural Optimization (DRASO), a principled way for injecting prior knowledge into the ASO framework. We also provide some analysis of the representations learned by our method.

1. Introduction

While supervised learning algorithms achieve impressive results on a variety of NLP tasks, they rely on the availability of labeled data. The application of other available resources to improve over existing supervised methods has been explored in semi-supervised learning. There are two primary sources of information for semi-supervised algorithms: unlabeled data and prior knowledge. Alternating Structual Optimization (ASO) (Ando & Zhang, 2005) is a semi-supervised learning technique based on unlabeled data, which has achieved considerable success in many important problems (Blitzer et al., 2006; Blitzer et al., 2007). ASO learns a new data representation by constructing and solving a multitask learning problem using unlabeled data. While ASO makes excellent use of unlabeled data, there is currently no way to encode prior information in learning the representations. For example, in the sentiment classification task, a short list of positive and negative words can be used to bootstrap learning (Turney, 2002).

In this work we seek to combine ASO with this type of prior knowledge. We present Declaratively Regularized ASO (DRASO), which favors learning representations that are consistent with some side information. DRASO combines both unlabeled data and prior knowledge to find a single representation of the data. This paper describes DRASO and shows that the representations learned for sentiment classification using side information can improve over a standard ASO representation.

2. DRASO

Given a number of related supervised learning problems, ASO learns a shared low dimensional representation of the data in order to minimize the empirical risk across the various tasks. Specifically, let the training set for task ℓ be $\{(\mathbf{x}_i^\ell, y_i^\ell)\}_{i=1}^{n_\ell}$. Given m such training sets, ASO learns a shared representation $\hat{\mathbf{\Phi}}$ and associated weight vectors $\hat{\mathbf{w}}_\ell$, $\hat{\mathbf{v}}_\ell$, $\ell = 1, ..., m$ by minimizing

Appearing in *Appearing in the Workshop on Prior Knowledge at the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008.

the loss over the training sets:

$$\left[\{\hat{\mathbf{w}}_\ell, \hat{\mathbf{v}}_\ell\}, \hat{\mathbf{\Phi}}\right] = \underset{\mathbf{w}_\ell, \mathbf{v}_\ell, \mathbf{\Phi}}{\operatorname{argmin}} \sum_{\ell=1}^{m} \left(\frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L\left((\mathbf{w}_\ell + \mathbf{\Phi}'\mathbf{v}_\ell)'\mathbf{x}_i^\ell, y_i^\ell\right) + \lambda ||\mathbf{w}_\ell||^2 \right) \quad \text{s.t. } \mathbf{\Phi}\mathbf{\Phi}' = I_{k\times k}.$$

The matrix $\mathbf{\Phi}$ is a shared transformation which maps a feature vector $\mathbf{x} \in \mathbb{R}^D$ to a low-dimensional vector in $\mathbb{R}^k$. Given $\mathbf{\Phi}$, $\mathbf{w}_\ell$, and $\mathbf{v}_\ell$, the prediction for an instance $\mathbf{x}^\ell$ is the linear function $(\mathbf{w}_\ell + \mathbf{\Phi}'\mathbf{v}_\ell)'\mathbf{x}^\ell$ where $\mathbf{w}_\ell$ is the weight vector applied to the original instance and $\mathbf{v}_\ell$ is the weight vector applied to the shared, low-dimensional represention, $\mathbf{\Phi}\mathbf{x}^\ell$.

Unfortunately, as written, the ASO criterion does not allow one to inject prior knowledge into the learned shared transformation $\mathbf{\Phi}$. For example, in the sentiment classification task, we may wish to represent the fact that presence of *excellent* or *superb* in a document express similar sentiment and hence a classifier should assign similar weights to the two features corresponding to the presence of these two words. To incorporate such declarative information, we suppose the existence of a prior knowledge graph which encodes knowledge about which features should be similarly correlated with the class labels in a "good" model. The nodes of the graph represent features and the edges represent feature similarities. The edges are weighted by the strength of similarity. These weights are encoded as a matrix $P \in R^{D\times D}$ with each entry $P_{ij} \geq 0$, $P_{ii} = 0$ and $\sum_j P_{ij} = 1$ for all i.

To enforce the similarity requirements, we replace the ridge regularization term with a penalty on the induced norm: $\mathbf{w}^{'} M \mathbf{w}$, where $M = (I - P)^{'}(I - P)$. This encourages features to be weighted similarly to the average of their neighbors' weights and is closely related to the LLE objective (Roweis & Saul, 2000; Sandler et al., 2008). The new optimization problem is then given as:

$$\left[\{\hat{\mathbf{w}}_\ell, \hat{\mathbf{v}}_\ell\}, \hat{\mathbf{\Phi}}\right] = \underset{\mathbf{w}_\ell, \mathbf{v}_\ell, \mathbf{\Phi}}{\operatorname{argmin}} \sum_{\ell=1}^{m} \left(\frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L\left((\mathbf{w}_\ell + \mathbf{\Phi}'\mathbf{v}_\ell)'\mathbf{x}_i^\ell, y_i^\ell\right) + \lambda \mathbf{w}^{'} M \mathbf{w} \right) \quad \text{s.t. } \mathbf{\Phi}M\mathbf{\Phi}' = I_{k\times k}.$$

We call this new objective DRASO, since the ASO objective is declaratively regularized. Solving for Φ yields a new eigenvalue problem, which can be solved efficiently (section 2.1).

2.1. Solving for Φ

Our main goal is to find the transformation Φ which we will use to create a new representation for the supervised problem. As in (Ando & Zhang, 2005), we can simplify the problem by making the change of variables $u_\ell = w_\ell + \Phi^{'} v_\ell$. This yields the optimization problem

$$\left[\{\hat{\mathbf{u}}_\ell, \hat{\mathbf{v}}_\ell\}, \hat{\mathbf{\Phi}}\right] = \underset{\mathbf{u}_\ell, \mathbf{v}_\ell, \mathbf{\Phi}}{\operatorname{argmin}} \sum_{\ell=1}^{m} \left(\frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L\left(\mathbf{u}_\ell^{'}\mathbf{x}_i^\ell, y_i^\ell\right) + \lambda(\mathbf{u}_\ell - \Phi^{'}\mathbf{v}_\ell)^{'} M (\mathbf{u}_\ell - \Phi^{'}\mathbf{v}_\ell) \right) \quad \text{s.t. } \mathbf{\Phi}M\mathbf{\Phi}' = I_{k\times k}.$$

Again following (Ando & Zhang, 2005), we can solve this problem using an alternating minimization technique. In the first step of the alternation, we fix Φ and $\mathbf{v}_\ell$ and solve for $\mathbf{u}$. As before, this step amounts to solving the decoupled linear predictions for each of the problems. In the second step, we fix $\mathbf{u}_\ell$ and solve for $\mathbf{v}_\ell$ and Φ. First we note that $\mathbf{v}_\ell$ has a closed form solution in terms of Φ and $\mathbf{u}_\ell$.

$$\left[\{\hat{\mathbf{v}}_\ell\}, \hat{\mathbf{\Phi}}\right] = \underset{\mathbf{v}_\ell, \mathbf{\Phi}}{\operatorname{argmin}} \sum_{\ell=1}^{m} \left(\lambda(\mathbf{u}_\ell - \Phi^{'}\mathbf{v}_\ell)^{'} M (\mathbf{u}_\ell - \Phi^{'}\mathbf{v}_\ell) \right) \quad \text{s.t. } \mathbf{\Phi}M\mathbf{\Phi}' = I_{k\times k}.$$

Solving for $\mathbf{v}_\ell$ in this quadratic form gives us $\mathbf{v}_\ell = \Phi M \mathbf{u}_\ell$. Now we can solve for the following minimization problem for Φ:

$$\left[\hat{\mathbf{\Phi}}\right] = \underset{\mathbf{\Phi}}{\operatorname{argmax}} \sum_{\ell=1}^{m} ||\Phi M \mathbf{u}_\ell||_2^2 \quad \text{s.t. } \mathbf{\Phi}M\mathbf{\Phi}' = I_{k\times k}.$$

Following (Ando & Zhang, 2005), we have that this problem is equivalent to the problem

$$\left[\hat{\mathbf{\Phi}}\right] = \underset{\mathbf{\Phi}}{\operatorname{argmax}} \operatorname{tr}\left(\mathbf{\Phi} M U U^{'} \mathbf{\Phi}\right) \quad \text{s.t. } \mathbf{\Phi}M\mathbf{\Phi}' = I_{k\times k}.$$

By looking at the first order conditions for the Lagrangian, we can see that the solutions have the form

$$MUU^{'}M\mathbf{\Phi} = \alpha M \mathbf{\Phi}$$

We can transform this generalized eigenvalue problem into one that is smaller and easier to manage if we let $\theta = U^{'} M\mathbf{\Phi}$. Now, right multiplying by $U^{'}$, we get:

$$U^{'} MU\theta = \alpha\theta$$

That is we can solve for the eigenvectors of the modified gram matrix (transformed via M). Now, we can substitute back into the original problem (noting that M is symmetric).

$$\begin{aligned} MUU^{'} M\mathbf{\Phi} &= \alpha M\mathbf{\Phi} \\ MU\theta &= \alpha M\mathbf{\Phi} \end{aligned}$$

Thus we have, $\mathbf{\Phi} = \frac{1}{\alpha} U\theta$.

3. Experimental Results

ASO and DRASO representations were compared on the sentiment classification task using Amazon book reviews from Blitzer et al. (2007). Auxiliary problems were selected using mutual information. Prior knowledge was obtained from SentiWordNet (Esuli & Sebastiani, 2006) by manually selecting 31 positive and 42 negative words from the top ranked positive and negative words in SentiWordNet. Each selected word was connected in graph P to its 10 nearest neighbors according to SentiWordNet rank.

The learned Φs were used to project 32,502 words into a two dimensional space (Figure 1). Words on the prior knowledge lists are indicated by squares (negative) and triangles (positive). Points are color coded based on their behavior in a large sample of labeled training data (13,391 instances) as red (positive), blue (negative) and grey (neutral). The figures indicate that list words clumped by ASO are separated by DRASO. Additionally, while pulling apart high-sentiment words, neutral words are left together. Finally, observe that additional points not on the list have been pulled as well, showing the effect of prior knowledge on new features. These results indicate that DRASO can incorporate prior information into ASO in a principled and effective way.

4. Related Work

The feature graph based additional regularization term in the DRASO objective is close in spirit to Fused Lasso (Tibshirani et al., 2005). However, there are crucial differences. Firstly, fused lasso assumes an ordering over the features while no such restriction is necessary in case of DRASO. Secondly, fused lasso imposes an L_1 penalty over differences in weights of consecutive features (assuming the features are ordered as mentioned above). In contrast, DRASO uses an L_2 norm and the regularization is imposed over immediate neighborhood rather than pairwise constraints. The L_1 penalty in (Tibshirani et al., 2005) prefers weights of linked features to be exactly same. However, in many problem domains (including the ones considered in this paper), it is desirable to have similar rather than identical weights.

The additional regularization term in the DRASO objective is similar to the one in Penalized Discriminant Analysis (PDA) (Hastie et al., 1995). While PDA performs standard classification, DRASO is focused on learning a new and more effective representation in ASO's multitask learning setting. The learned representation could in turn be used as additional features in a standard classifier, which is currently being investigated (Section 5).

5. Conclusion

In this paper we have presented DRASO, which extends ASO by adding a regularization term. This additional term makes it possible to inject valuable prior knowledge into the ASO framework. We have shown that while solving for Φ, incorporation of the additional regularization term results in an eigenvalue problem (different from ASO) which can be solved efficiently. We have also presented experimental evidence demonstrating effectiveness of DRASO over ASO.

For future work, we are considering applications to learning tasks for which ASO has performed well. For many of these tasks, prior knowledge can be added through existing resources or through the use of unsupervised methods to infer relations between features. We are also investigating under what conditions prior knowledge can improve over labeled data alone.

Acknowledgment

We would like to thank the anonymous reviewers for helpful feedback. This work is supported in part by NSF IIS-0513778.

References

Ando, R. K., & Zhang, T. (2005). A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research (JMLR)*, *6*, 1817–1853.

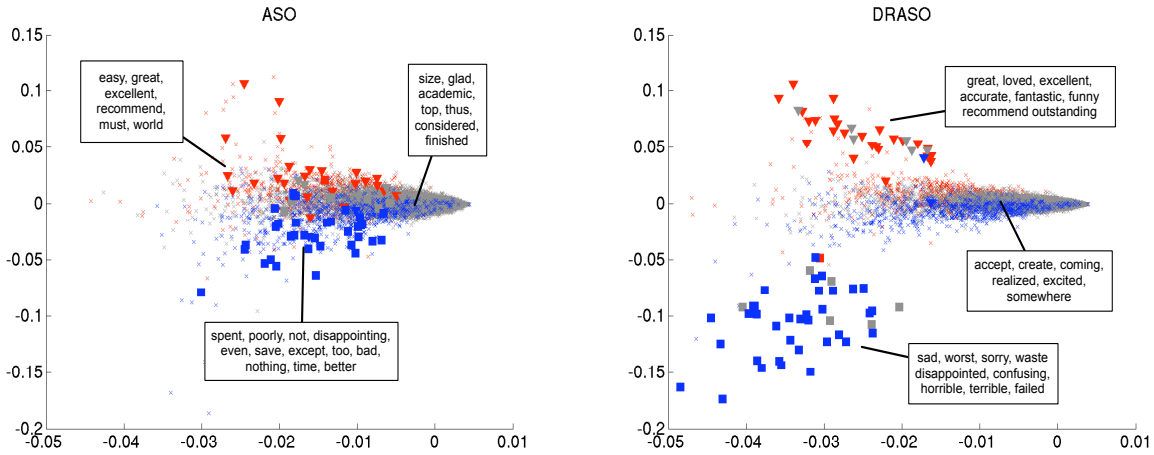

Figure 1. ASO and DRASO projections of 32,502 words into a two-dimensional space. Squares (negative) and triangles (positive) indicate prior knowledge words. Polarity of features as measured from labeled data is indicated by blue (negative), red (positive) and grey (neutral). Some of the features are annotated to demonstrate the effects of the projection.

Blitzer, J., Dredze, M., & Pereira, F. (2007). Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. *Association for Computational Linguistics (ACL)*.

Blitzer, J., McDonald, R., & Pereira, F. (2006). Domain adaptation with structural correspondence learning. *Empirical Methods in Natural Language Processing (EMNLP)*.

Esuli, A., & Sebastiani, F. (2006). Sentiwordnet: A publicly available lexical resource for opinion mining. *Proceedings of LREC*, 417–422.

Hastie, T., Buja, A., & Tibshirani, R. (1995). Penalized discriminant analysis. *Annals of Statistics*, *23*, 73–102.

Roweis, S., & Saul, L. (2000). Nonlinear Dimensionality Reduction by Locally Linear Embedding.

Sandler, T., Blitzer, J., & Ungar, L. H. (2008). Learning with locally linear feature regularization. *Snowbird Learning Workshop*.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., & Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society Series B*, *67*, 91–108.

Turney, P. (2002). Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *Association for Computational Linguistics (ACL)*.

Zhang, T. (2004). Solving large scale linear prediction problems using stochastic gradient descent algorithms. *ACM International Conference Proceeding Series*.

Bayesian Modeling of Dependency Trees Using Hierarchical Pitman-Yor Priors

Hanna M. Wallach WALLACH@CS.UMASS.EDU
Department of Computer Science, University of Massachusetts, Amherst, MA 01003 USA

Charles Sutton SUTTON@CS.BERKELEY.EDU
Computer Science Division, University of California, Berkeley, CA 94720 USA

Andrew McCallum MCCALLUM@CS.UMASS.EDU
Department of Computer Science, University of Massachusetts, Amherst, MA 01003 USA

1. Introduction

Recent work on hierarchical priors for n-gram language modeling [MacKay and Peto, 1995, Teh, 2006, Goldwater et al., 2006] has demonstrated that Bayesian methods can be used to reinterpret well-known non-Bayesian techniques for smoothing sparse counts. However, sparse counts are not unique to language modeling—they are ubiquitous throughout NLP—and the same ideas may be used to reinterpret and enhance other non-Bayesian NLP models, thereby extending the reach of Bayesian methods in natural language.

In addition to word order—the focus of n-gram language modeling—natural language also exhibits complex syntactic structures. Dependency trees are a useful way of representing these kinds of structures. Dependency trees encode relationships between words and their sentence-level, syntactic modifiers by representing a sentence as a tree with a node for each word. The parent of each word is the word that it most directly modifies. Despite modeling different kinds of structure, generative models of dependency trees are similar to n-gram language models in that they both decompose the probability of a sentence into a product of probabilities of individual words given some (typically sparse) word-based context. In n-gram models, the context is the preceding words, while in dependency modeling it is the word's parent and sometimes its siblings. Thus, while the actual contexts used by the models are different, the underlying idea—that contexts consist of nearby words—is the same. The models do differ in two important ways, however. First, while all information (word identities and order) is observed in an n-gram model, dependency models require inference of the latent structure of each dependency tree. Second, unlike n-gram modeling, in which trigrams are smoothed with bigrams and so on, the choice of context reductions for dependency models is less obvious and must be decided by the modeler.

In this paper, we describe two hierarchical Bayesian models for dependency trees. First, we show that Eisner's classic generative dependency model [1996] can be substantially improved by (a) using a hierarchical Pitman-Yor process as a prior over the distribution over dependents of a word, and (b) sampling the model hyperparameters (section 3). These changes alone yield a significant increase in parse accuracy over Eisner's model. Second, we present a Bayesian dependency parsing model in which latent state variables mediate the relationships between words and their dependents. This model clusters dependencies into states using a similar approach to that employed by Bayesian topic models when clustering words into topics (section 4). The inferred states have a syntactic flavor and lead to modestly improved accuracy when substituted for part-of-speech tags in the parsing model.

2. Background

In this section, we briefly review the hierarchical Pitman-Yor process and its application to n-gram language modeling. The Pitman-Yor process [Pitman and Yor, 1997] has three parameters: a base measure $\boldsymbol{m}$, a concentration parameter α, and a discount parameter $0 \leq \epsilon < 1$. In an n-gram language model the probability of word w in the context of $\boldsymbol{h}$ (a sequence of $n-1$ words) is $\phi_{w|\boldsymbol{h}}$. Letting $\rho(\boldsymbol{h})$ be the reduction of $\boldsymbol{h}$, obtained by dropping the left-most word, each probability vector $\boldsymbol{\phi}_{\boldsymbol{h}} = \{\phi_{w|\boldsymbol{h}}\}$ can be given a Pitman-Yor prior, with parameters $\boldsymbol{m}_{\rho(\boldsymbol{h})}$, α_{n-1} and ϵ_{n-1}. The base measure $\boldsymbol{m}_{\rho(\boldsymbol{h})}$ is shared by all contexts $\boldsymbol{h}'$ with reduction $\rho(\boldsymbol{h}') = \rho(\boldsymbol{h})$. The effects of

$P(s_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n)$	$P(w_n \mid s_n, s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, d_n)$	$P(c_n \mid s_n, w_n)$
$s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n$	$s_n, s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, d_n$	s_n, w_n
$s_{\pi(n)}, s_{\sigma(n)}, d_n$	$s_n, s_{\pi(n)}, d_n$	$s_n,$
$s_{\pi(n)}, d_n$	$s_n,$	

Table 1. Contexts (in order) used by Eisner for estimating probabilities.

using a Pitman-Yor prior are best explained in terms of drawing a new observation from the predictive distribution over words given $\boldsymbol{h}$, obtained by integrating out $\boldsymbol{\phi}_{\boldsymbol{h}}$: If the observation is the first to be drawn, it is instantiated to the value of a new "internal" draw from $\boldsymbol{m}_{\rho(\boldsymbol{h})}$. Otherwise, it is instantiated to the value of an existing internal draw, with probability proportional to the number of observations previously "matched" to that draw minus ϵ_{n-1}, or to the value of a new internal draw, with probability proportional to α_{n-1}. The Pitman-Yor process may be used hierarchically—i.e., $\boldsymbol{m}_{\rho(\boldsymbol{h})}$ may be given a Pitman-Yor prior, with parameters $\boldsymbol{m}_{\rho(\rho(\boldsymbol{h}))}$, α_{n-2} and ϵ_{n-2}, and integrated out. Similarly for $\boldsymbol{m}_{\rho(\rho(\boldsymbol{h}))} \ldots \boldsymbol{m}_{\emptyset}$. This yields a hierarchy of Pitman-Yor processes encompassing all context reductions. The internal draws at one level are treated as observations by the next level up, and there is path from each observation to top-level uniform base measure $\boldsymbol{u}$ via the internal draws. The observation counts in the predictive distribution are effectively smoothed with higher-level counts, determined by the number of observations (or lower-level internal draws) matched to each internal draw in the hierarchy. The hierarchical Pitman-Yor process was applied to n-gram language modeling by Teh [2006] and Goldwater et al. [2006].

For real-world data, the number of internal draws at each level and the paths from the observations to the top-level base measure $\boldsymbol{u}$ are unknown. Since these quantities determine the counts used in the predictive distribution, they must be inferred using either Gibbs sampling or an approximate inference scheme.

Bayesian n-gram language modeling was first explored by MacKay and Peto [1995], who drew connections between non-Bayesian interpolated language models and hierarchical Dirichlet priors. Teh [2006] and Goldwater et al. [2006] showed that using a hierarchical Pitman-Yor process prior as described above leads to a model of which Kneser-Ney smoothing is a special case.

3. A Hierarchical Pitman-Yor Dependency Model

In this section, we describe the first of our Bayesian dependency parsing models. This model is best explained by starting with a reinterpretation of Eisner's dependency model [1996] from a Bayesian perspective. Eisner's model generates sentences using a parent-outward process. Each parent generates a sequence of children starting in the center and moving outward to the left and then similarly to the right. Conditioned on the parent, the sequence of children in each direction is a first order Markov chain. The probability of a sentence consisting of words $\boldsymbol{w}$, with corresponding part-of-speech tags $\boldsymbol{s}$, case values $\boldsymbol{c}$ (see below) and tree $\boldsymbol{t}$, generated according to this process, is

$$\begin{aligned} P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}) = \\ \prod_n & P(s_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n) \\ & P(w_n \mid s_n, s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, d_n) \\ & P(c_n \mid s_n, w_n). \end{aligned} \tag{1}$$

where d_n is the direction of w_n with respect to its parent, $\pi(n)$ is the position of w_n's parent, $\sigma(n)$ the position of w_n's immediately preceding sibling (moving outward from w_n's parent in direction d_n), and $y(n)$ is the position of w_n's final child. The case c_n of each word w_n may be one of four values: lower, upper, mixed, or first capitalized word in the sentence.

Eisner estimates each probability in equation 1 from training data $\mathcal{D}$ (tagged, cased sentences and their trees) by interpolating between probability estimates computed using various reduced conditioning contexts. The complete set of conditioning contexts for each variable (i.e., tag, word, case) are shown in table 1.

Alternatively, however, equation 1 can be rewritten as

$$\begin{aligned} P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}) = \\ \prod_n & \theta_{s_n \mid s_{\pi(n)} w_{\pi(n)} c_{\pi(n)} s_{\sigma(n)} d_n} \\ & \phi_{w_n \mid s_n, s_{\pi(n)} w_{\pi(n)} c_{\pi(n)} d_n} \, \psi_{c_n \mid s_n w_n} \end{aligned} \tag{2}$$

where $\boldsymbol{\theta}_{s'w'c's''d}$ is the distribution over part-of-speech tags for the context consisting of parent tag s', parent word w', parent case value c', sibling tag s'', and direction d. Similarly, $\boldsymbol{\phi}_{ss'w'c'd}$ is the distribution over words for the context defined by tag s, parent tag s', parent word w', parent case value c', and direction d. Finally, $\boldsymbol{\psi}_{sw}$ is the distribution over case values for the context consisting of tag s and word w. Eisner's interpolation method is then equivalent to giving each

probability vector a hierarchical Dirichlet prior—e.g.,

$$\boldsymbol{\theta}_{s'w'c's''d} \sim \mathrm{Dir}\left(\boldsymbol{\theta}_{s'w'c's''d} \,|\, \alpha_2, \boldsymbol{m}_{s's''d}\right) \quad (3)$$

$$\boldsymbol{m}_{s's''d} \sim \mathrm{Dir}\left(\boldsymbol{m}_{s's''d} \,|\, \alpha_1, \boldsymbol{m}_{s'd}\right) \quad (4)$$

$$\boldsymbol{m}_{s'd} \sim \mathrm{Dir}\left(\boldsymbol{m}_{s'd} \,|\, \alpha_0, \boldsymbol{u}\right) \quad (5)$$

with $\alpha_2 = \alpha_1 = 3$ and $\alpha_0 = 0.5$ (the parameter values used by Eisner). Under these hierarchical priors, the predictive distributions given data $\mathcal{D}$ (computed as described by MacKay and Peto [1995]) are identical to the interpolated probabilities used by Eisner.

This Bayesian reinterpretation of Eisner's model has two advantages: Firstly, the concentration parameters may be sampled, rather than fixed to some particular value. Secondly, it is also possible to use priors other than the hierarchical Dirichlet distribution—for example, a hierarchical Pitman-Yor process prior:

$$\boldsymbol{\theta}_{s'w'c's''d} \sim \mathrm{PY}\left(\boldsymbol{\theta}_{s'w'c's''d} \,|\, \alpha_2, \boldsymbol{m}_{s's''d}, \epsilon_2\right) \quad (6)$$

$$\boldsymbol{m}_{s's''d} \sim \mathrm{PY}\left(\boldsymbol{m}_{s's''d} \,|\, \alpha_1, \boldsymbol{m}_{s'd}, \epsilon_1\right) \quad (7)$$

$$\boldsymbol{m}_{s'd} \sim \mathrm{PY}\left(\boldsymbol{m}_{s'd} \,|\, \alpha_0, \boldsymbol{u}, \epsilon_0\right). \quad (8)$$

Priors for $\boldsymbol{\phi}_{ss'w'c'd}$ and $\boldsymbol{\psi}_{sw}$ can similarly be defined using the context reductions shown in table 1.

3.1. Inference

Given the above hierarchical Pitman-Yor dependency parsing model and a training corpus $\mathcal{D}$, consisting of tagged, cased sentences and their trees, there are two tasks of interest: sampling hyperparameters (αs and ϵs) and inferring trees for unseen test sentences.

Having inferred a set of internal draws for $\mathcal{D}$, typical concentration and discount parameters can be determined using slice sampling [Neal, 2003]. Then, given a set of hyperparameter values U, the parents for all words in a test sentence can be jointly sampled using an algorithm that combines dynamic programming with the Metropolis-Hastings method. The resultant algorithm is similar to that of Johnson et al. [2007a,b] for unlexicalized probabilistic context-free grammars.

For each sentence $\boldsymbol{w}$, a proposal tree $\boldsymbol{t}'$ is sampled from the following distribution using a dynamic program based on Eisner's $O(N^3)$ parsing algorithm [1]:

$$P(\boldsymbol{t}' \,|\, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, U)$$

$$\simeq P(\boldsymbol{t}' \,|\, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \{\hat{\boldsymbol{\theta}}_{s'w'c's''d}, \hat{\boldsymbol{\phi}}_{ss'w'c'd}, \hat{\boldsymbol{\psi}}_{sw}\}, U) \quad (9)$$

$$\propto P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}' \,|\, \{\hat{\boldsymbol{\theta}}_{s'w'c's''d}, \hat{\boldsymbol{\phi}}_{ss'w'c'd}, \hat{\boldsymbol{\psi}}_{sw}\}, U), \quad (10)$$

where $\mathcal{D}_{\setminus \boldsymbol{w},\boldsymbol{s},\boldsymbol{c},\boldsymbol{t}}$ is the corpus excluding the tagged, cased sentence of interest and its previously sampled tree $\boldsymbol{t}$. The probability vectors $\hat{\boldsymbol{\theta}}_{s'w'c's''d}$, $\hat{\boldsymbol{\phi}}_{ss'w'c'd}$ and $\hat{\boldsymbol{\psi}}_{sw}$ are the predictive distributions over tags, words and case values given $\mathcal{D}_{\setminus \boldsymbol{w},\boldsymbol{s},\boldsymbol{c},\boldsymbol{t}}$ and the current set of internal draws and paths. The proposal tree $\boldsymbol{t}'$ is sampled from an approximation to the true posterior since sampling from the true posterior is not possible.

Having generated a proposal tree $\boldsymbol{t}'$, it is accepted with probability given by the minimum of 1 and

$$\frac{P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}' \,|\, \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, U)}{P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t} \,|\, \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, U)} \frac{P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t} \,|\, \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, \hat{\Theta}, \hat{\Phi}, \hat{\Psi}, U)}{P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}' \,|\, \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, \hat{\Theta}, \hat{\Phi}, \hat{\Psi}, U)}, \quad (11)$$

where $\hat{\Theta} = \{\hat{\boldsymbol{\theta}}_{w',s',c',s'',d}\}$, $\hat{\Phi} = \{\hat{\boldsymbol{\phi}}_{s,w',s',c',d}\}$ and $\hat{\Psi} = \{\hat{\boldsymbol{\psi}}_{w,s}\}$. If $\boldsymbol{t}'$ is rejected, then the previously sampled tree $\boldsymbol{t}$ is kept as the current assignment for $\boldsymbol{w}$.

3.2. Results

Dependency parsing models are typically evaluated by computing parse accuracy—i.e., the percentage of parents correctly identified. The hierarchical Pitman-Yor dependency model was used to parse the Wall Street Journal sections of the Penn Treebank [Marcus et al., 1993]. To facilitate comparison with other dependency parsing algorithms, the standard train/test split was used (sections 2–21 for training and section 23 for testing), and parse accuracies were computed using the maximum probability trees. The Penn Treebank training sections consist of 39,832 sentences, while the test section consists of 2,416 sentences. Words that occur in the test data but not in training and words that occur once in training data but never in the test data were replaced with UNSEEN types. data, while tags for the test data were inferred using a standard part-of-speech tagger [Ratnaparkhi, 1996]. [2] Punctuation words were excluded from all accuracy calculations.

We compared four different priors: (i) Hierarchical Dirichlet with fixed concentration parameters, set to the values used by Eisner. When used with an approximate inference scheme known as the maximal path assumption, this model variant is identical to Eisner's model; (ii) Hierarchical Dirichlet with slice-sampled concentration parameters; (iii) Pitman-Yor with fixed concentration parameters, set to the values used by Eisner, and fixed discount parameters set to 0.1; (iv) Pitman-Yor with slice-sampled hy-

[1] Details are omitted due to space restrictions.

[2] The generative nature of the dependency parser means that it is possible to sample part-of-speech tags for test sentences at the same time as sampling their trees. However, this is computationally expensive and gives very similar performance to using tags from Ratnaparkhi's tagger.

		Path Assumption	
		Maximal	Minimal
Dirichlet	fixed α values [Eisner, 1996]	80.7	80.2
Dirichlet	sampled α values	84.3	84.1
Pitman-Yor	fixed α and ϵ values	83.6	83.7
Pitman-Yor	sampled α and ϵ values	85.4	**85.7**

Table 2. Parse accuracy of the hierarchical Pitman-Yor dependency model.

perparameters. For each prior, two approximate inference schemes—the *maximal and minimal path assumptions* [Cowans, 2006]—were compared. For the model variants with sampled hyperparameters, fifty slice sampling iterations was sufficient for convergence.

Parse accuracies are shown in table 2. These results show that (a) using a hierarchical Pitman-Yor prior and (b) sampling hyperparameters both give considerable performance improvements over a hierarchical Dirichlet dependency parser with fixed concentration parameters and the maximal path assumption (equivalent to Eisner's model). Using a hierarchical Pitman-Yor prior and sampling hyperparameters yield orthogonal improvements of 3%–5% each over Eisner's parser. Together, these two modeling choices yield a 26% error reduction. The differences in parse accuracy between the approximate inference schemes (maximal and minimal path assumptions) are not significant.

The accuracies for the model variant that is equivalent to Eisner's dependency model (hierarchical Dirichlet prior, maximal path assumption, fixed concentration parameters) are lower than those reported in Eisner's original work [Eisner, 1996]. This is because Eisner's results were obtained using an extensively filtered data set with only 400 test sentences (e.g., sentences with conjunctions were discarded). In the time since Eisner's model was published a different train/test split has become standard, and the results reported in table 2 were computed on the now-standard split.

Although state-of-the-art dependency models, such as the discriminative maximum-margin method of McDonald [2006], achieve higher parse accuracy, it is possible that further enhancements to the Pitman-Yor dependency model would yield similar results while retaining the benefits of a generative model. Possible enhancements include a detailed consideration of contexts and reductions, aggregation across multiple tree samples, Gibbs sampling the internal draws and paths done by Teh [2006], and using a letter-based language model as a top-level base measure [Cowans, 2006].

4. A "Syntactic Topic" Dependency Model

One advantage of a generative approach to dependency modeling is that other latent variables can be incorporated into the model. To demonstrate this, we present a second Bayesian dependency model with latent state variables that mediate the relationships between words and their dependents. These variables result in a syntactic clustering of parent–child dependencies. This model can be considered to be a dependency-based analogue of the syntactic component from the syntax-based topic model of Griffiths et al. [2005]. The models differ in their underlying structure, however: In the dependency model in this section, the underlying structure is a tree that combines both words and unobserved syntactic states; in Griffiths et al.'s model, the structure is a simply a linear chain over latent states. This difference means that there are two kinds of latent information that must be inferred in the dependency-based model: The structure of each dependency tree and the identities of the latent states. In Griffiths et al.'s model, only the latter need be inferred.

4.1. Model

The generative process underlying the model in this section is similar to that of the model presented in the previous section. The main difference is that instead of generating a child directly, a parent word first generates a syntactic state, which then generates the child word. Additionally, for computational efficiency, the children in each direction are independent conditioned on their parent. The probability of an untagged sentence $\boldsymbol{w}$ with latent states $\boldsymbol{s}$ and tree $\boldsymbol{t}$ is given by

$$P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{t}) = \prod_n \theta_{s_n \,|\, w_{\pi(n)}} \, \phi_{w_n \,|\, s_n}, \qquad (12)$$

where $\boldsymbol{\theta}_{w'}$ is the distribution over latent states for parent word w', and $\boldsymbol{\phi}_s$ is the distribution over child words for latent state s. Parent words are collapsed down to the latent state space and children are generated on the basis of these states. As a result, the clusters induced by the latent states exhibit syntactic properties and can be thought of as "syntactic topics"—specialized

distributions over words with a syntactic flavor. Each of the probability vectors in equation 12 is given a single-level Dirichlet prior as shown below:

$$\boldsymbol{\theta}_{w'} \sim \text{Dir}\left(\boldsymbol{\theta}_{w'} \,|\, \alpha, \boldsymbol{m}\right) \quad (13)$$

$$\boldsymbol{\phi}_{s} \sim \text{Dir}\left(\boldsymbol{\phi}_{s} \,|\, \beta, \boldsymbol{u}\right) \quad (14)$$

The base measure $\boldsymbol{m}$ and concentration parameter α for the prior over over $\boldsymbol{\theta}_{w'}$ are optimized together.

4.2. Inference

Given a training corpus $\mathcal{D} = \{\boldsymbol{w}, \boldsymbol{t}\}$ consisting of untagged sentences and their corresponding trees, there are two tasks of interest: Sampling latent states for $\mathcal{D}$, and sampling states and trees for unseen test sentences. States for a training sentence are sampled using Gibbs sampling. Each state s_n is sampled from the conditional distribution for that state given all other state assignments, and the training data:

$$\begin{aligned} P(s_n = k \,|\, \{\boldsymbol{w}\}, \{\boldsymbol{s}\}_{\setminus n}, \{\boldsymbol{t}\}, U) \propto \\ P(w_n \,|\, s_n = k, \{\boldsymbol{s}\}_{\setminus n}, \{\boldsymbol{w}\}_{\setminus n}, \{\boldsymbol{t}\}) \\ P(s_n = k \,|\, \{\boldsymbol{s}\}_{\setminus n}, \{\boldsymbol{w}\}_{\setminus n}, \{\boldsymbol{t}\}), \end{aligned}$$

where the subscript "$\setminus n$" denotes a quantity that excludes data from the n^{th} position in the corpus.

Given a set of training sentences and trees and a single sample of training states, the trees and states for unseen test sentences may be sampled using an augmented version of the dynamic program in section 3.1.

4.3. Results

The true dependency trees and words in Penn Treebank sections 2–21 were used to obtain a single sample of latent states. These states, trees and words were then used to sample states and trees for the 2,416 sentences in Penn Treebank section 23. Some example states or "syntactic topics" are shown in table 3. Each column in each row consists of the words most likely to be generated by a particular state. The states exhibit a good correspondence with parts-of-speech, but are more finely grained. For example, the states in the first and third columns in the top row both correspond to nouns. However, the first contains job titles, while the third contains place names. The states in the fourth and fifth columns in the top row both correspond to verbs. However, the fourth contains transitive past-tense verbs, while the fifth contains present-tense verbs. This kind of specificity indicates that these states are likely to be beneficial in other tasks where part-of-speech tags are typically used, such as named entity recognition and machine translation.

	Type of Tree	
	Sampled	Max. Prob.
POS tags	55.3	63.1
50 states	59.2	63.8
100 states	60.0	64.1
150 states	60.5	64.7
200 states	60.4	64.5

Table 4. Parse accuracy of the "syntactic topic" model on the Penn Treebank (standard train/test split). As a baseline, the latent states are fixed to part-of-speech tags. Results for sampled trees are averaged over ten samples.

The quality of these "syntactic topics" was measured by using them in place of part-of-speech tags in supervised parsing experiments. The latent state dependency model (with 50, 100, 150 and 200 states) was compared with an equivalent model in which the states were fixed to true part-of-speech tags for both training and test data. These results are shown in table 4. Using the sampled states gives an improvement in parse accuracy of approximately 5% for sampled trees and an improvement of 1.6% for the most probable trees. Although this is a modest improvement, it is a clear quantitative indication that the discovered states do indeed capture syntactically meaningful information.

5. Related Work

There has been much recent interest in nonparametric Bayesian models for PCFGs with latent variables [Liang et al., 2007, Petrov et al., 2006, Finkel et al., 2007], as well as general inference and learning frameworks for Bayesian PCFGs [Johnson et al., 2007a,b]. While previous work has focused on latent variables, state splitting, and inference in unlexicalized PCFG models, the dependency models presented in this paper are lexicalized. Lexicalization, in which parent–child statistics are incorporated into the model, is an important technique for building high-accuracy parsing models, although state-splitting and discriminative models can obtain similar benefits. Unfortunately, lexicalized models are much more likely to suffer from sparsity problems. As a result, smoothing is critical—as reflected in the structure of our hierarchical prior. Previous nonparametric Bayesian models for grammars have not concentrated on smoothing issues.

6. Conclusions

In this paper, we introduced a new generative dependency parsing model based on the hierarchical Pitman-Yor process. Using this model, we showed that the

president	year	u.s.	made	is	in
director	years	california	offered	are	on
officer	months	washington	filed	was	,
chairman	quarter	texas	put	has	for
executive	example	york	asked	have	at
head	days	london	approved	were	with
attorney	time	japan	announced	will	and
manager	weeks	canada	left	had	as
chief	period	france	held	's	by
secretary	week	britain	bought	would	up
10	would	more	his	ms.	sales
8	will	most	their	mrs.	issues
1	could	very	's	who	prices
50	should	so	her	van	earnings
2	can	too	and	mary	results
15	might	than	my	lee	stocks
20	had	less	your	dorrance	rates
30	may	and	own	linda	costs
25	must	enough	'	carol	terms
3	owns	about	old	hart	figures

Table 3. Example states inferred by the "syntactic topic" model. Each column in each row shows the words most likely to be generated as children by states inferred from Treebank dependency trees. (From a model with 150 states.)

performance of Eisner's generative dependency parsing model can be significantly improved by using a hierarchical Pitman-Yor prior and by sampling model hyperparameters. On the Penn Treebank data, this leads to a 26% reduction in parsing error over Eisner's model. We also presented a second Bayesian dependency model, in which the local dependency distributions are mediated by latent variables that cluster parent–child dependencies. Not only do the inferred latent variables look like finer-grained parts-of-speech, they result in modestly improved parse accuracy when substituted for part-of-speech tags in the model. Our future work will include models that combine dependency trees with both semantic and syntactic topics.

7. Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, in part by DoD contract #HM1582-06-1-2013, and in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0427594. Any opinions, findings, conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

P. J. Cowans. *Probabilistic Document Modeling*. PhD thesis, University of Cambridge, 2006.

J. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *COLING-96*, 1996.

J. Finkel, T. Grenager, and C. Manning. The infinite tree. In *ACL*, 2007.

S. Goldwater, T. Griffiths, and M. Johnson. Interpolating between types and tokens by estimating power-law generators. In *NIPS 18*. 2006.

T. Griffiths, M. Steyvers, D. Blei, and J. Tenenbaum. Integrating topics and syntax. In *NIPS 17*. 2005.

M. Johnson, T. Griffiths, and S. Goldwater. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *NAACL*, 2007a.

M. Johnson, T. Griffiths, and S. Goldwater. Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In *NIPS 19*. 2007b.

P. Liang, S. Petrov, M. I. Jordan, and D. Klein. The infinite PCFG using hierarchical Dirichlet processes. In *EMNLP/CoNLL*, 2007.

D. J. C. MacKay and L. C. B. Peto. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1995.

M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 1993.

R. McDonald. *Discriminative Training and Spanning Tree Algorithms for Dependency Parsing*. PhD thesis, University of Pennsylvania, 2006.

R. M. Neal. Slice sampling. *Annals of Statistics*, 2003.

S. Petrov, L. Barrett, R. Thibaux, and D. Klein. Learning accurate, compact, and interpretable tree annotation. In *ACL*, 2006.

J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 1997.

A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *EMNLP*, 1996.

Y. W. Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *ACL*, 2006.

Using Participant Role in Multiparty Meetings as Prior Knowledge for Nonparametric Topic Modeling

Songfang Huang S.F.HUANG@ED.AC.UK
Steve Renals S.RENALS@ED.AC.UK
The Centre for Speech Technology Research, University of Edinburgh, Edinburgh, EH8 9LW, United Kingdom

Abstract

In this paper we introduce our attempts to incorporate the participant role information in multiparty meetings for document modeling using the hierarchical Dirichlet process. The perplexity and automatic speech recognition results demonstrate that the participant role information is a promising prior knowledge source to be combined with language models for automatic speech recognition and interaction modeling for multiparty meetings.

1. Introduction

In recent years there has been growing research interest in the automatic speech recognition (ASR) for multiparty meetings, which is of essential importance for the subsequent meeting processing such as content analysis, summarisation, discourse analysis, and information retrieval. In this paper, we consider an improved language model (LM) in a state-of-the-art large vocabulary ASR system for meetings, based on the prior knowledge of participant roles. More specifically, we estimate the word distribution over the role of each participant, i.e., $P(w|r)$, and use this as unigram marginals to adapt a conventional n-gram LM.

The AMI and AMIDA (http://www.amiproject.org) projects are dedicated to the development of technologies to enhance the recognition and interpretation of interactions between people in multiparty meetings (Renals et al., 2007). The AMI Meeting Corpus collected by the AMI project consists of 100 hours of multimodal meeting recordings with comprehensive annotations at a number of different levels. About 70% of the corpus was elicited using a design scenario, in which the participants play the roles of employees, i.e., project manager (PM), marketing expert (ME), user interface designer (UI), and industrial designer (ID), in an electronics company that decides to develop a new type of television remote control. Our intuition is that, since different participants play different roles, there may be a different word distribution, and in turn different dominant words, specific to each role. For example, we expect a project manager is more likely to speak words relating to the coordination of meetings, i.e., meeting, project, or present, while a user interface designer may favor words on interaction mediums like screen, voice, or speech.

Topic models have received much attention in the machine learning community, which follows the "bag-of-words" assumption, i.e., words in a document are exchangeable. In this paper we attempt to incorporate the participant role as prior knowledge in topic models, by assigning role information to exchangeable words in a document. This could be achieved within the flexible framework of topic models, by introducing an additional observed variable for the role into the graphical model. By assuming that each role has a mixture distribution over the latent topics, we could infer the topic distribution specific to each role. We could further estimate $P(w|r)$ for each role r by integrating out the latent topics. Moreover, incorporating the role in topic models enables not only the *document* modeling, but also the *interaction* modeling in meetings.

An alternative approach to modeling the relationship between the participant role information and lexical words is to directly estimate the conditional probability $P(w|r)$ based on the co-occurrence statistics of roles and words, using the maximum likelihood principle. As a comparison to the probabilistic topic models, we also introduce in this paper a deterministic approach to modeling roles and words, by regarding the role as an additional feature (factor) of lexical words in an MLE-based LM.

Appearing in *the Workshop on Prior Knowledge for Text and Language Processing* at ICML/UAI/COLT 2008, Helsinki, Finland, 2008.

2. Modeling Approaches

We consider two modeling approaches to the estimation of $P(w|r)$: one is a hierarchical Bayesian model using the hierarchical Dirichlet process (HDP) as the prior, and the other is a factored approach using the factored language model (FLM).

2.1. Hierarchical Bayesian Model

The hierarchical Dirichlet process (Teh et al., 2006) is a nonparametric generalization of latent Dirichlet allocation (LDA), which extends the standard LDA model to infinite and hierarchical topic modeling.

Conversational speech consists of sequences of utterances, which do not comprise well-defined documents. We used the following procedure to obtain documents: for each scenario meeting, first align all the words in it along a common timeline; then for each sentence/segment, collect those non-stop words belonging to a window of length L as the document, by backtracing from the end time of the sentence/segment. The role that has been assigned to the most of words in the window is selected as the role for that document. We use a moving window with $L = 20$ seconds over the sequences of words to obtain documents.

We incorporate the participant role by extending the 2-level HDP (Teh et al., 2006) in Figure 1(A) to a third level, as shown in Figure 1(B), role-HDP. An DP G_r is assigned for each of the four roles (PM,ME,UI,ID), which then served as the parent DP (the base probability measure) in the HDP hierarchy for all those DPs corresponding to documents belonging to that role.

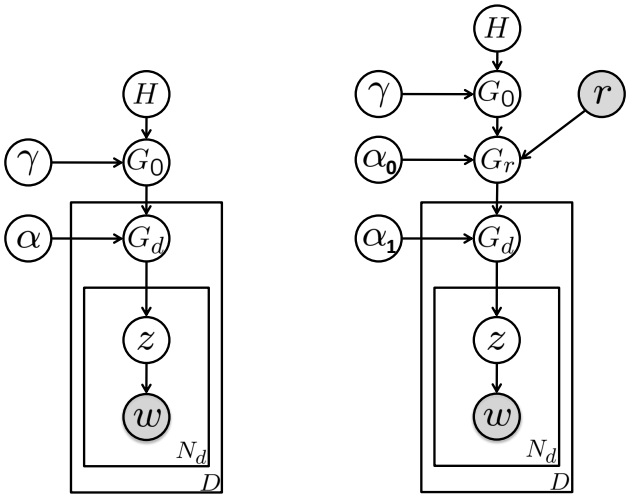

Figure 1. The graphical model depictions for (A) 2-level HDP, and (B) role-HDP.

2.2. Factored Language Model

One straightforward method for modeling words and roles is to use the maximum likelihood estimation based on the co-occurrences of words w and the role information r, i.e., training a bigram-like model $P(w|r) = \text{Count}(w, r)/\text{Count}(r)$. More generally, we can use a factored language model (Bilmes & Kirchhoff, 2003) to model words and role deterministically. The FLM, initially developed to address the language modeling problems faced by morphologically rich or inflected languages, is a generalization of standard n-gram language models, in which each word w_t is decomposed into a bundle of K word-related features (called *factors*), $w_t \equiv f_t^{1:K} = \{f_t^1, f_t^1, \ldots, f_t^K\}$. Factors may include the word itself. Each word in an FLM is dependent not only on a single stream of its preceding words, but also on additional parallel streams of factors. Combining with interpolation or generalized parallel backoff (GPB) (Bilmes & Kirchhoff, 2003) strategies, multiple backoff paths may be used simultaneously.

We exploit two factors for word w at time t: the word w_t itself and the corresponding role r_t, as shown in Figure 2. All the words in a sentence share a common role, i.e., $r_t = r_{t-1} = \ldots = r_1$ in Figure 2. We use a simple backoff strategy, for example, by moving from the model $P(w_t|r_t)$ directly down to the unigram model $P(w_t)$. We refer this model to the role-FLM.

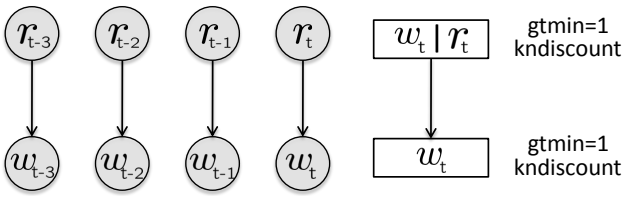

Figure 2. The graphical model representation and backoff path for role-FLM.

2.3. Combination with n-gram LMs

As in (Kneser et al., 1997), we use the dynamic unigram marginals $P(w|r)$, from either the role-HDP or the role-FLM, for LM adaptation:

$$P_{\text{adapt}}(w|h) = P_{\text{back}}(w|h) \cdot \left(\frac{P(w|r)}{P_{\text{back}}(w)} \right)^{\mu} / z(h) \quad (1)$$

where h is the history of w, $P_{\text{back}}(w|h)$ the baseline n-gram, $P_{\text{adapt}}(w|h)$ the adapted n-gram, and $z(h)$ a normalisation factor. For the role-HDP, $P(w|r) \approx \sum_{k=1}^{K} \phi_{kw} \cdot \theta_{dk}$ with $\boldsymbol{\phi}_k$ estimated during training and remaining fixed in testing, while $\boldsymbol{\theta}_d$ are document-dependent (and in turn are role-dependent because $\boldsymbol{\theta}_d$

are derived from G_r) and thus are calculated dynamically for each test document.

3. Experiments and Results[1]

3.1. Empirical Experiment

We first carried out some empirical analyses for the HDP and the role-HDP. The HDP was implemented as an extension to the SRILM toolkit[2]. We trained the HDP and the role-HDP models using different values ($k = 1, \ldots, 100$) for the initial number of topics. We used uniform distribution for H, i.e., $H_w = 1/W$. All models were trained using the fold-2−4 of the AMI scenario meetings, with a fixed size vocabulary of 7,910 words, by the Markov Chain Monte Carlo (MCMC) sampling method. The concentration parameters were sampled using the auxiliary variable sample scheme in (Teh et al., 2006). We ran 3,000 iterations to burn-in, then collected 10 samples from the posteriors to calculate the unigram perplexity on the fold-1 testing data, with the sample step of 5.

Figure 3 shows the perplexity results, from which we can see that the role-HDP produced better results than the HDP. Our understanding for the improvement is that by using the role prior knowledge, documents with the same role share the strengths in the HDP framework. In addition, we show in Figure 4 the top two topics for each role. It is interesting to find out that each role has some specific topics with high probabilities, while they also tend to interact with each other on some common topics, i.e., the button topic appears with high probability for all the four roles in Figure 4.

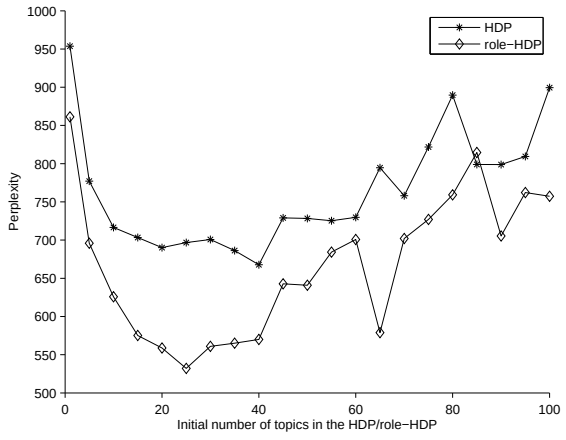

Figure 3. The perplexity results for the HDP/role-HDP.

[1]Some of the results here were appearing in another paper by the authors in (Huang & Renals, 2008).

[2]`http://www.speech.sri.com/projects/srilm`

PM		ME		UI		ID	
(29) 0.28	(14) 0.13	(20) 0.27	(14) 0.16	(14) 0.25	(19) 0.15	(14) 0.17	(6) 0.16
DESIGN	BUTTON	REMOTE	BUTTON	BUTTON	RECOGNITION	BUTTON	CHIP
MEETING	BUTTONS	CONTROL	BUTTONS	BUTTONS	SPEECH	BUTTONS	SIMPLE
PROJECT	CHANNEL	LOOK	CHANNEL	CHANNEL	VOICE	CHANNEL	ADVANCED
MINUTES	SCREEN	MARKET	SCREEN	SCREEN	L_C_D_	SCREEN	REGULAR
USER	SCROLL	CONTROLS	SCROLL	SCROLL	SCREEN	SCROLL	REMOTE
INTERFACE	VOLUME	MEAN	VOLUME	VOLUME	MICROPHONE	VOLUME	L_C_D_
PRESENT	MENU	EASY	MENU	MENU	CONTROLLER	MENU	BATTERY
PRODUCT	L_C_D_	PRODUCT	L_C_D_	L_C_D_	REMOTE	L_C_D_	SPEAKER
DESIGNER	REMOTE	DESIGN	REMOTE	REMOTE	PROBLEM	REMOTE	EXPENSIVE
LOOK	WHEEL	BUTTONS	WHEEL	WHEEL	TECHNOLOGY	WHEEL	INFRARED
START	MEAN	FANCY	MEAN	MEAN	COFFEE	MEAN	STATION
SURE	CHANNELS	IMPORTANT	CHANNELS	CHANNELS	SPEAKER	CHANNELS	SAMPLE
BIT	CONTROL	PERCENT	CONTROL	CONTROL	COST	CONTROL	BATTERIES
GUESS	PRESS	FIND	PRESS	PRESS	CONTROL	PRESS	COST
THANK	KIND	LOT	KIND	KIND	SYSTEM	KIND	CONTROL

Figure 4. The example topics for the four roles using the role-HDP.

3.2. ASR Experiment

To further investigate the effectiveness of employing the role as prior knowledge for topic modeling, we performed ASR experiments on multiparty meetings. We used part of the AMI Meeting Corpus for our experiments. There are 138 scenario meetings in total, of which 118 were used for training and the other 20 for testing (about 11 hours). The procedure and parameters used to train the HDP/role-HDP were the same as those used in Section 3.1, except that we used a different split-up of the AMI scenario meetings for ASR.

We trained two baseline LMs: the first one used the Fisher conversational telephone speech data (fisher-03-p1+p2), and the second used three datasets from the AMI training data, the Fisher, and the Hub-4 broadcast news data (hub4-lm96). The two baseline LMs were trained with standard parameters using SRILM: trigrams, cut-off value of 2 for trigram counts, modified Kneser-Ney smoothing, interpolated model. A common vocabulary with 56,168 words was used for the two LMs, which has 568 out-of-vocabulary (OOV) words for the AMI test data.

We investigated the effectiveness of the adapted LMs based on topic and role information from meetings on a practical large vocabulary ASR system. The AMI-ASR system (Hain & et al., 2007) was used as the baseline system. We began from the lattices for the whole AMI Meeting Corpus, generated by the AMIASR system using a trigram LM trained on a large set of data coming from Fisher, Hub4, Switchboard, webdata, and various meeting sources including AMI. We then generated 500-best lists from the lattices for each utterance. We adapted the two baseline LMs (Fisher and AMI+Fisher+Hub4) using Equation (1) according to the unigram marginals from the role-FLM, the HDP, and the role-HDP respectively. For the HDP, we used $P(w|d) \approx \sum_{k=1}^{K} \phi_{kw} \cdot \theta_{dk}$ as the unigram marginals, i.e., the difference from $P(w|r)$ by the role-HDP is that in the HDP $\boldsymbol{\theta}_d$ are only document-dependent

Table 1. The %WER results of ASR experiments using adapted LMs on the AMI scenario meetings.

LMs	SUB	DEL	INS	WER
Fisher	22.7	11.4	5.8	39.9
role-FLM-adapted	22.5	11.1	5.9	39.5
HDP-adapted	22.2	11.3	5.6	39.1
role-HDP-adapted	22.3	11.3	5.6	39.2
AMI+Fisher+Hub4	21.6	11.1	5.4	38.2
role-FLM-adapted	21.4	10.9	5.6	37.9
HDP-adapted	21.2	11.1	5.3	37.6
role-HDP-adapted	21.2	11.1	5.3	37.5

but not role-dependent. The topics were extracted by the HDP/role-HDP models based on the previous ASR outputs, using a moving document window with a length of 10 seconds. Three adapted LMs together with the baseline LM were then used to rescore the 500-best lists with a common language model weight of 14 (the same as for lattice generation) and no word insertion penalty. The adapted LM was destroyed after it was used to rescore the current N-best lists.

Table 1 shows the word error rate (WER) results. We can see that both the HDP and the role-HDP adapted LMs yield significant reductions ($p < 0.01$ according to a matched-pair significance test [3]) in WER, comparing to the baseline LMs. Although the role-FLM adapted LMs also reduce the WER, this deterministic approach is not as effective as the probabilistic topic modeling by introducing a latent variable – topic. However, there is no significant difference between the HDP and the role-HDP.

4. Discussion

Although the approach we used here to incorporate the role as prior knowledge for topic modeling is straightforward, the preliminary experiments demonstrated not only the better perplexity and WER results, but also the ability for modeling specific topic distributions for each role. This suggests some future work on the use of role information as prior knowledge is worth further investigation in the following aspects:

Probabilistic. The fact that we only assign one role for each document implies that we may lose some information, because there are potentially multiple roles for a document by using a moving window to obtain documents. Therefore, it is better to use a more probabilistic way, for example, each document is regarded as a multinomial distribution over roles, and each role a multinomial distribution over topics. Moreover, this helps to model the interactions between roles.

Observed vs. Latent. Depending on whether we treat the role variable as observed or latent, we can exploit the role as prior knowledge for topic modeling (as in this paper), or use other information to infer the role for each document. The latter is useful for modeling the human interactions in multiparty meetings.

Application. Even if we observed reductions in perplexity, it is not trivial to *transfer* the advantage of using the role prior knowledge for topic modeling to real applications such as on language modeling for automatic speech recognition in meetings. We observed no significant difference between the HDP and the role-HDP for ASR. We are interested in either a method of explicitly conditioning on the role for language modeling, or an approach to tightly combining topic models and n-gram models.

[3] `http://www.icsi.berkeley.edu/speech/faq/signiftest.html`

Acknowledgments

This work is jointly supported by the Wolfson Microelectronics Scholarship and the European IST Programme Project FP6-033812 (AMIDA). This paper only reflects the authors' views and funding agencies are not liable for any use that may be made of the information contained herein.

References

Bilmes, J. A., & Kirchhoff, K. (2003). Factored language models and generalized parallel backoff. *Proceedings of HLT/NACCL* (pp. 4–6).

Hain, T., & et al. (2007). The AMI system for the transcription of speech in meetings. *Proc. of ICASSP'07*.

Huang, S., & Renals, S. (2008). Modeling topic and role information in meetings using the hierarchical Dirichlet process. *Proc. of Machine Learning for Multimodal Interaction (MLMI'08)*.

Kneser, R., Peters, J., & Klakow, D. (1997). Language model adaptation using dynamic marginals. *Proc. of Eurospeech*. Rhodes.

Renals, S., Hain, T., & Bourlard, H. (2007). Recognition and interpretation of meetings: The AMI and AMIDA projects. *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding*.

Teh, Y., Jordan, M., Beal, M., & Blei, D. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, *101*, 1566–1581.

Exponential family sparse coding with application to self-taught learning with text documents

Honglak Lee HLLEE@CS.STANFORD.EDU
Rajat Raina RAJATR@CS.STANFORD.EDU
Alex Teichman TEICHMAN@CS.STANFORD.EDU
Andrew Y. Ng ANG@CS.STANFORD.EDU
Stanford University, Stanford, CA 94305 USA

Abstract

Sparse coding is an unsupervised learning algorithm for finding concise, slightly higher-level representations of an input, and has been successfully applied to self-taught learning (Raina et al., 2007), where the goal is to use unlabeled data to help on a supervised learning task, even if the unlabeled data cannot be associated with the labels of the supervised task. However, sparse coding uses a Gaussian noise model and a quadratic loss function, and thus performs poorly if applied to binary valued, integer valued, or other non-Gaussian data, such as text. Drawing on ideas from Generalized linear models (GLMs), we present a generalization of sparse coding to learning with data drawn from any exponential family distribution (such as Bernoulli, Poisson, etc). This gives a method that we argue is much better suited to model other data types than Gaussian. We present an efficient algorithm for solving the optimization problem defined by this model. We also show that the new model results in significantly improved self-taught learning performance when applied to text data.

1. Introduction

We consider the "self-taught learning" problem, in which we are given just a few labeled examples for a classification task, and also large amounts of unlabeled data that is only mildly related to the task (Raina et al., 2007; Weston et al., 2006). Specifically, the unlabeled data may not share the class labels or arise from the same distribution. For example, given only a few labeled examples of webpages about "Baseball" or "Football", along with access to a large corpus of unrelated webpages, we might want to accurately classify new baseball/football webpages. The ability to use such easily available unlabeled data has the potential to greatly reduce the effect of data sparsity, and thus greatly improve performance on labeling or tagging applications in language processing.

Our approach uses the unlabeled data to learn a high-level representation of the inputs, and then using this representation to provide features for classification. Raina et al. demonstrate such a method for domains such as image classification, using the "sparse coding" model (Olshausen & Field, 1996). In this model, given real-valued vectors $x \in \mathbb{R}^k$ as inputs, we attempt to learn a large set of basis vectors $b_1, b_2, \ldots, b_n \in \mathbb{R}^k$ such that the input can be represented as a linear combination of only a few basis vectors: i.e., $x \approx \sum_j b_j s_j$, where s_j is the weight (or "activation") for basis vector b_j, and most s_j values are zero (or, the vector s is *sparse*). Informally, the activation vector s used to reconstruct an input x often captures the few "most important" patterns in x. For example, when this model is applied to images, the basis vectors learnt by the model represent various edge patterns, and thus s captures the edges present in an image input x. Indeed, when the activations s are used as features in a standard supervised learning algorithm (such as an SVM), the generalization performance is often better than when using the raw input x as features.

The sparse coding algorithm for learning basis vectors is based on a Gaussian noise model for input x: $P(x|b,s) = \mathcal{N}(\sum_j b_j s_j, \sigma^2 I)$, where σ^2 is fixed. A sparse prior $P(s) \propto \prod_j \exp(-\beta|s_j|)$ is assumed on the activations to penalize nonzero activations. Then, given unlabeled examples $\{x^{(1)}, x^{(2)}, \ldots\}$ and the corresponding activations $\{s^{(1)}, s^{(2)}, \ldots\}$, good basis vectors are estimated by solving the MAP optimization problem, which is equivalent to solving:[1]

$$\min_{\{b_j\},\{s^{(i)}\}} \quad \tfrac{1}{2\sigma^2} \textstyle\sum_i \|x^{(i)} - \sum_{j=1}^n b_j s_j^{(i)}\|^2 + \beta \sum_{i,j} |s_j^{(i)}|$$
$$\text{subject to} \qquad \|b_j\|^2 \le c,\ \forall j = 1, ..., n.$$

[1] Following previous work, we constrain the norm of each basis vector b_j to make the problem well-posed.

This optimization problem is convex separately in the b and s variables (though not jointly convex). The problem can be solved efficiently by alternating minimization over b and s variables (Lee et al., 2007). Finally, given learnt basis vectors b, the features for a new input example x are derived by estimating: $\arg\max_s P(s|x,b) = \arg\max_s P(x|b,s)P(s)$.

2. Self-taught learning for text data

The probabilistic model used in sparse coding assumes that the input vectors x are real-valued, and that they can be well described using a Gaussian noise model. However, this is an inappropriate assumption for text data, which is often represented as a binary "bag-of-words" vector $x \in \{0,1\}^k$, where the i-th feature is 1 if the i-th word in our vocabulary occured in the document, or as a word-counts vector $x \in \{0,1,2,\dots\}^k$, where the i-th feature represents the number of times the i-th word occured in the document. In either case, the input vectors are very poorly modeled by a continuous Gaussian distribution (which could take fractional, or negative values). It is thus hardly surprising that when sparse coding is applied to a self-taught learning task for text, it only leads to small gains in accuracy, even with huge amounts of unlabeled data; in some cases, it can even hurt performance.

To address this problem, in this paper we generalize the Gaussian probabilistic model behind sparse coding in a principled way to "exponential family" of distributions. This class of distributions is large enough to include most commonly used distributions (including the Gaussian, Bernoulli and Poisson distributions among others), but also guarantees crucial properties that make efficient learning and inference possible (e.g., the maximum likelihood learning problem is convex for any distribution in the family) (McCullagh & Nelder, 1989). We call our model *exponential family sparse coding*, and to differentiate it from the previous model, we henceforth call that model *Gaussian sparse coding*.

3. Exponential family sparse coding

Given a text document (input vector) x, we use the standard exponential family model: $P(x|b,s) = h(x)\exp(\eta^T T(x) - a(\eta))$ with the natural parameter $\eta = \sum_j b_j s_j$. Here the functions h, T and a specify the exact exponential family distribution being used (e.g., $h(x) = e^{-\|x\|^2/2}/(2\pi)^{k/2}$, $T(x) = x$, $a(\eta) = \eta^T\eta/2$ lead to a Gaussian distribution with covariance I). This includes the Gaussian sparse coding model as a special case, and we can now use a Poisson distribution if the input is integer-valued, or a Bernoulli distribution if the input is binary.

With this generative model, the basis vectors b can again be learnt from unlabeled data by solving the MAP optimization problem, or equivalently:[2]

$$\begin{aligned}\min_{B,\{s^{(i)}\}} \quad & \textstyle\sum_i \Big(-\log h(x^{(i)}) - {s^{(i)}}^T B^T T(x^{(i)}) + a(Bs^{(i)})\Big) \\ & \qquad +\beta \textstyle\sum_{i,j} |s_j^{(i)}| \\ \text{s.t.} \quad & \qquad \|b_j\|^2 \le c,\ \forall j = 1,...,n. \end{aligned} \tag{1}$$

Inspite of the generalization, this problem is still convex separately in b and s (though not jointly).[3] This again suggests an alternating minimization procedure iterating the following two steps till convergence: (i) fix the activations s, and compute the optimal basis vectors B; and, (ii) fix these basis vectors B, and compute the optimal activations s.

Step (i) involves a constrained optimization problem over B with a differentiable objective function. We can thus apply projective gradient descent updates, where at each iteration we perform a line search along the direction of the (negative) gradient, projecting onto the feasible set before evaluating the objective function during the line search. In our case, the projection operation is especially simple: we just need to rescale each basis vector to have norm c if its norm is greater than c. In our experiments, we find that such a projective gradient descent scheme is sufficiently fast for basis learning. We thus focus now on the algorithm for computing the optimal activations in Step (ii).

Step (ii) computes the optimal activation s given fixed basis vectors. The resulting problem involves a non-differentiable L_1-regularized objective function, for which several sophisticated algorithms have been developed, including specialized interior point methods (Koh et al., 2007) and quasi-Newton methods (Andrew & Gao, 2007; Yu et al., 2008). When used for computing activations with 1000 basis vectors, these methods find the optimal solution in a few seconds *per unlabeled example*. Since we often need to solve for tens of thousands of unlabeled examples repeatedly in the inner loop of the overall alternating minimization procedure, these solvers turn out to be too slow for high-dimensional problems with many basis vectors. We now present an alternative, efficient algorithm for computing the activations.

4. Computing optimal activations

We first note that since the optimal values for the activation vectors $s^{(i)}$ do not depend on each other, and can be optimized separately, it is sufficient to consider

[2] We use matrix notation $B = [b_1 b_2 \dots b_n]$.

[3] This follows from the fact that $a(\eta)$ must be convex in η (McCullagh & Nelder, 1989).

the following optimization problem for a single input x and its activation vector s:

$$\min_s \quad \ell(s) + \beta\|s\|_1 \tag{2}$$

where s corresponds to a vector of activations, and $\ell(s)$ is a specific convex function of s.

In the case of Gaussian sparse coding, $\ell(s)$ is simply a quadratic function, and the optimization problem is an L_1-regularized least squares problem that can be solved efficiently (Efron et al., 2004; Lee et al., 2007). This suggests an iterative algorithm for general exponential family distributions: at each iteration, we compute a local quadratic approximation $\hat{\ell}(s)$ to the function $\ell(s)$, and optimize the objective function $\hat{\ell}(s) + \beta\|s\|_1$ instead.[4] Using a similar insight, Lee et al. proposed the IRLS-LARS algorithm for the case of L_1-regularized logistic regression, using Efron et al.'s LARS algorithm in the inner loop to solve the approximated problem.

This method can be applied to other L_1-regularized optimization problems for which a local quadratic approximation can be efficiently computed. Indeed, for the case of the L_1-regularized exponential family in Equation (1), we can show that the local quadratic approximation at a point s is given by:

$$\hat{\ell}(s') \quad = \quad \left\|\Lambda^{1/2}Bs' - \Lambda^{1/2}z\right\|^2 \tag{3}$$

where $\Lambda_{ii} = a''\left((Bs)_i\right)$ for a diagonal matrix Λ, and $z = \Lambda^{-1}(T(x) - a'(Bs)) + Bs$.[5]

We further note that if the objective function $\ell(s)$ is reasonably approximated by a quadratic function, the solutions to the successive quadratic approximations should be close to each other. However, the LARS algorithm used in IRLS-LARS cannot be initialized at an arbitrary point, and thus has to rediscover the solution from scratch while solving each successive approximation. On the other hand, the "feature-sign search" algorithm (originally proposed in the context of Gaussian sparse coding) can be initialized at an arbitrary point (Lee et al., 2007), and can thus potentially solve the successive approximations much faster. We propose to use the feature-sign search algorithm to optimize each quadratic approximation.

The final algorithm, which we call IRLS-FS, is described below. The algorithm is guaranteed to converge to the global optimum in a finite number of iterations. (Proof similar to IRLS-LARS.)

[4]This procedure is an instance of a more general method that is sometimes called Iteratively Reweighted Least Squares (IRLS) in the literature (Green, 1984).

[5]To show that this is the local quadratic approximation to $\ell(s)$, it suffices to show that this has the same function value, gradient and Hessian at s. Indeed, we have $\nabla\ell = \nabla\hat{\ell} = -B^T T(x) + B^T a'(Bs)$, and $\nabla^2\ell = \nabla^2\hat{\ell} = B^T\Lambda B$.

Algorithm 1: IRLS-FS algorithm

Input: $B \in \mathbb{R}^{k\times n}$: matrix, $x \in \mathbb{R}^k$: vector.
Initialize $s := \vec{0}$.
while stopping criterion is not satisfied **do**
  $\Lambda_{ii} := a''\left((Bs)_i\right)$ (for diagonal matrix Λ)
  $z := \Lambda^{-1}(T(x) - a'(Bs)) + Bs$.
  Initializing feature-sign search at s, compute:
    $\hat{s} := \arg\min_{s'} \left\|\Lambda^{1/2}Bs' - \Lambda^{1/2}z\right\|^2 + \beta\|s'\|_1$
  Set $s := (1-t)s + t\hat{s}$, where t is found by a backtracking line-search that minimizes the original objective function in problem (1). (See Boyd & Vandenberghe, 2004 for details)
end while

5. Computational Efficiency

We compare the IRLS-FS algorithm against state-of-the-art algorithms for optimizing the activations, focusing on the case of binary sparse coding (i.e., $x \in \{0,1\}^k$). This case is especially interesting because this leads to an L_1-regularized logistic regression problem.[6] This problem is of general interest (e.g., see Ng, 2004), and customized algorithms have also been developed for it.

We compare the algorithm with four recent algorithms: the IRLS-LARS algorithm (Lee et al., 2006) and the l1-logreg interior point method (Koh et al., 2007) specifically for logistic regression, and the OWL-QN (Andrew & Gao, 2007) and SubLBFGS (Yu et al., 2008) algorithms for L_1-regularized convex optimization problems.[7] All algorithms were evaluated on nine L_1-regularized logistic regression problems, which arise when computing activations for text data. The sparsity parameter β was set to produce roughly 20 nonzero activations per example on average. We measured the running time taken by each algorithm to converge within a specified tolerance of the optimal

[6]We note that in each L_1-regularized optimization problem produced by exponential family sparse coding, the number of "features" is equal to the number of basis vectors used, but is *independent* of the dimensionality of inputs x in the original problem. For example, when applied to text, the number of "features" is equal to the number of basis vectors, but is independent of the number of words in the vocabulary, which could be large.

[7]Baseline algorithms: Lee et al. show that IRLS-LARS outperforms several previous algorithms, including grafting (Perkins & Theiler, 2003), SCGIS (Goodman, 2004), GenLasso (Roth, 2004) and Gl1ce (Lokhorst, 1999). IRLS-FS, IRLS-LARS and l1-logreg were implemented in Matlab, and OWL-QN and SubLBFGS were compiled in C++ with optimization flags. Code for all baselines was obtained from the respective authors.

Dataset	Small1	Small2	Small3	Medium1	Medium2	Medium3	Large1	Large2	Large3
IRLS-LARS	4.6	4.9	4.3	12.8	12.5	13.2	1131	1270	1214
l1-logreg	18.3	18.9	17.7	181	188	185	3277	3101	3013
OWL-QN	7.1	7.0	10.3	27.1	31.4	25.6	1018	739	906
SubLBFGS	33.0	22.3	23.1	101	142	57.2	1953	2717	1627
IRLS-FS	**2.5**	**2.3**	**2.2**	**5.3**	**5.5**	**5.4**	**117**	**108**	**109**

Table 1: Total running time in seconds for computing activations for 50 input examples for 9 problems (one per column). There are 3 problems each of 3 different sizes, and they are labeled Small1 to Small3, Medium1 to Medium3, or Large1 to Large3 based on the size of the problem. The Small problems had 200 basis vectors and input dimension 369, Med problems had 600 basis vectors and dimension 369, and Large problems had 1000 basis vectors and dimension 3891.

Dataset	Col	Alon	Duln	Duer	Arr	Mad	Hep	Spf	Prom	Wbc	Ion	Spl	Spc	Spam
IRLS-LARS	2.1	3.3	6.2	35.6	2.2	25.6	0.5	5.0	2.1	5.0	3.5	18.3	2.6	57.8
l1-logreg	18.3	16.8	13.6	14.4	34.8	509	1.0	3.0	2.0	3.8	2.7	12.8	2.0	**37.1**
OWL-QN	27.4	29.4	16.9	79.6	7.7	634	**0.1**	3.4	**0.4**	13.4	**1.9**	**7.1**	**0.9**	39.3
SubLBFGS	114	80.8	60.5	311	24.3	261	0.7	9.3	2.7	14.4	4.5	13.4	2.1	43.0
IRLS-FS	**1.9**	**1.9**	**2.5**	**7.1**	**1.5**	**14.0**	0.3	**2.3**	1.3	**2.9**	2.0	10.4	1.9	50.8

Table 2: Total running time in seconds for 50 trials of learning parameters of various L_1-regularized logistic regression benchmarks. The sparsity parameter β was picked to optimize generalization error of the resulting classifier. The datasets are ordered from left-to-right by increasing fraction of nonzero parameter values at the optimal solution (e.g., the problem Col had 0.2% nonzeros, Mad: 3.2%, Hep: 26.3%, Spam: 66.7%).

objective value.[8]

Table 1 shows the running times computed over 50 trials. IRLS-FS outperforms the other algorithms on this task, showing that it is well-suited to exponential family sparse coding. When a large number of basis vectors are used (while keeping the number of nonzero activations fixed), IRLS-FS can be 5 to 7 times faster than the best baseline algorithm.

This poses the question: can IRLS-FS be a useful algorithm for general L_1-regularized optimization problems (not necessarily ones generated by the sparse coding problem)? We compare the algorithms above on 14 moderate-size benchmark classification datasets, and apply L_1-regularized logistic regression to them.[9] The value of β on each benchmark was picked to optimize the generalization error of the resulting logistic regression classifier; unlike the earlier experiment, β was not set explicitly to obtain sparse solutions. Table 2 shows the running time of the algorithms to compute the optimal parameters. IRLS-FS outperforms the other algorithms on 8 out of 14 of these benchmark datasets; as expected, it performs best when the optimal parameters have few nonzero values.

[8]Details: Since IRLS-LARS solves the dual or Lasso version of our problem (i.e., with a constraint C on the L_1 norm of the activations rather than a penalty β), we follow Lee et al.'s method of using the KKT conditions to convert between the constraint value C and the equivalent penalty β for that problem. We ran all algorithms until they reached an objective value of $(1+\epsilon)f^{opt}$ where f^{opt} is the optimal objective value (we used $\epsilon = 10^{-6}$).

[9]These datasets were used to evaluate IRLS-LARS and were obtained from the authors (Lee et al., 2006).

6. Self-taught learning for text documents

The model presented in this paper generalizes Gaussian sparse coding. It is also closely related to exponential family PCA (Collins et al., 2001), which corresponds to setting the sparsity penalty β to zero, and additionally constraining the basis matrix B to have orthogonal columns. We now show that the exponential family sparse coding model can produce better self-taught learning performance than either of these previous methods.

We apply our algorithm to two self-taught learning problems in text classification: one using binary-valued input vectors $x \in \{0,1\}^k$, and another using integer-valued (word count) vectors $x \in \{0,1,2,\dots\}^k$.

We constructed five different webpage classification problems, and a newsgroup classification problem. We used 470,000 unlabeled news articles (from the Reuters corpus) to learn basis vectors according to the binary and Poisson sparse coding models.[10] Table 3 gives ex-

[10]Details: The webpage classification problems were created using subcategories of the Arts, Business, Health, Recreation and Sports categories of the DMOZ hierarchy. Each of them consisted of 10 separate binary classification problems over a 500 word vocabulary, with stopword removal and stemming. The newsgroup classification problem consisted of 10 binary classification problems constructed using the 20 newsgroups dataset. We used 600 basis vectors, and picked β to achieve roughly 10% nonzero activations. For learning, we used stochastic updates with mini-batches of 2000 randomly sampled examples, and assumed that the basis vectors had converged when the objective function did not decrease for 10 consecutive mini-batch iterations.

free	share	pharmaceut	estat	subscrib
market	exchange	drug	real	online
polici	stock	product	sale	servic
power	secur	share	loss	server
peopl	commiss	stock	loan	databas
paint	actor	novel	audio	singer
pictur	actress	literari	video	pop
portrait	film	poet	dvd	releas
museum	comedi	fiction	digit	fan
rule	star	univers	softwar	busi

Table 3: Ten examples of basis vectors trained on the Reuters data using Poisson sparse coding. Each group of five words were the most highly active words (i.e., had the highest weight) for some basis vector. Thus, each set of five words is judged to be highly correlated with each other. The top half contains basis vectors over the vocabulary for the Business category, the bottom half for the Arts category.

amples of basis vectors that were learned by applying Poisson sparse coding. Basis vectors appeared to encode related words and capture various "topics."

Using the learnt basis vectors, we computed features for each classification task using the binary and Poisson sparse coding model. We call our model "ExpSC" and compare against several baselines: the raw words themselves ("Raw"), Gaussian sparse coding ("GSC"), exponential family PCA with the same binary or Poisson exponential family assumption ("ExpPCA"), and also Latent Dirichlet Allocation (LDA), a widely-known topic model for text documents (Blei et al., 2002). All baselines (except the raw features) were trained using the same unlabeled data as our model. We also consider combinations of the raw word features with the other types of features (e.g., "Raw+ExpSC" indicates a combination of the raw features and the ExpSC features). All features were then evaluated using standard supervised-learning classifiers over 100 trials each for 4, 10 and 20 training examples.[11]

Figure 1 shows the classification results obtained for various training set sizes. The left 6 figures show results for Poisson sparse coding with varying numbers of training examples on the x-axis; the right 6 figures show results for binary sparse coding. Poisson and binary sparse coding reduce error significantly over the raw features in five and four out of six tasks respectively. The results for Poisson sparse coding are particularly striking, showing 20-30% error reduction in some cases.

Table 4 shows the average test error over all problems for the binary and Poisson case. The exponential family sparse coding features alone frequently outperform the other features, and produce slightly better results when used in combination with the raw features (ExpSC+Raw). The other three methods for using unlabeled data (GSC, ExpPCA, LDA) perform poorly in many cases.

[11] To evaluate the dependency of our results on the choice of classifier, we first evaluated many generic classifiers including SVM, Gaussian discriminant analysis (GDA), kernel dependency estimation (KDE), KNN, decision trees, etc; then, we chose the three best classifiers (GDA, KDE, and SVM) for the raw bag-of-words features. We report average results of the best performing classifier for each feature. We picked the β value used for computing the features by cross-validation. We verified that tuning the classifier hyperparameters by cross-validation does not significantly affect the results.

Discussion

In this paper, we present a general method for self-taught learning, that extends previous models in a natural way. The extensions can still be solved efficiently using the IRLS-FS algorithm, which we show to be an efficient algorithm for medium-sized L_1-regularized learning problems with sparse optimal solutions. We have shown that our model achieves better self-taught learning performance than Gaussian sparse coding or exponential family PCA. Overall, our results suggest that exponential family sparse coding can learn high-level representations of documents from unlabeled data, and that this prior knowledge is useful in document classification problems. We believe this model could be applied more generally to other language problems, including information retrieval and word sense disambiguation, where large amounts of unlabeled data are available.

Acknowledgments

We give warm thanks to Roger Grosse for useful comments. We also thank Kwangmoo Koh, Su-In Lee, Jin Yu, and Galen Andrew for providing their code. This work was supported by the DARPA transfer learning program under contract number FA8750-05-2-0249, and by ONR under award number N00014-06-1-0828.

References

Andrew, G., & Gao, J. (2007). Scalable training of L_1-regularized log-linear models. *ICML*.

Blei, D., Ng, A. Y., & Jordan, M. (2002). Latent dirichlet allocation. *NIPS*.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Collins, M., Dasgupta, S., & Schapire, R. E. (2001). A generalization of principal component analysis to the exponential family. In *NIPS*.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *Ann. Stat.*, *32*, 407–499.

Goodman, J. (2004). Exponential priors for maximum entropy models. *ACL*.

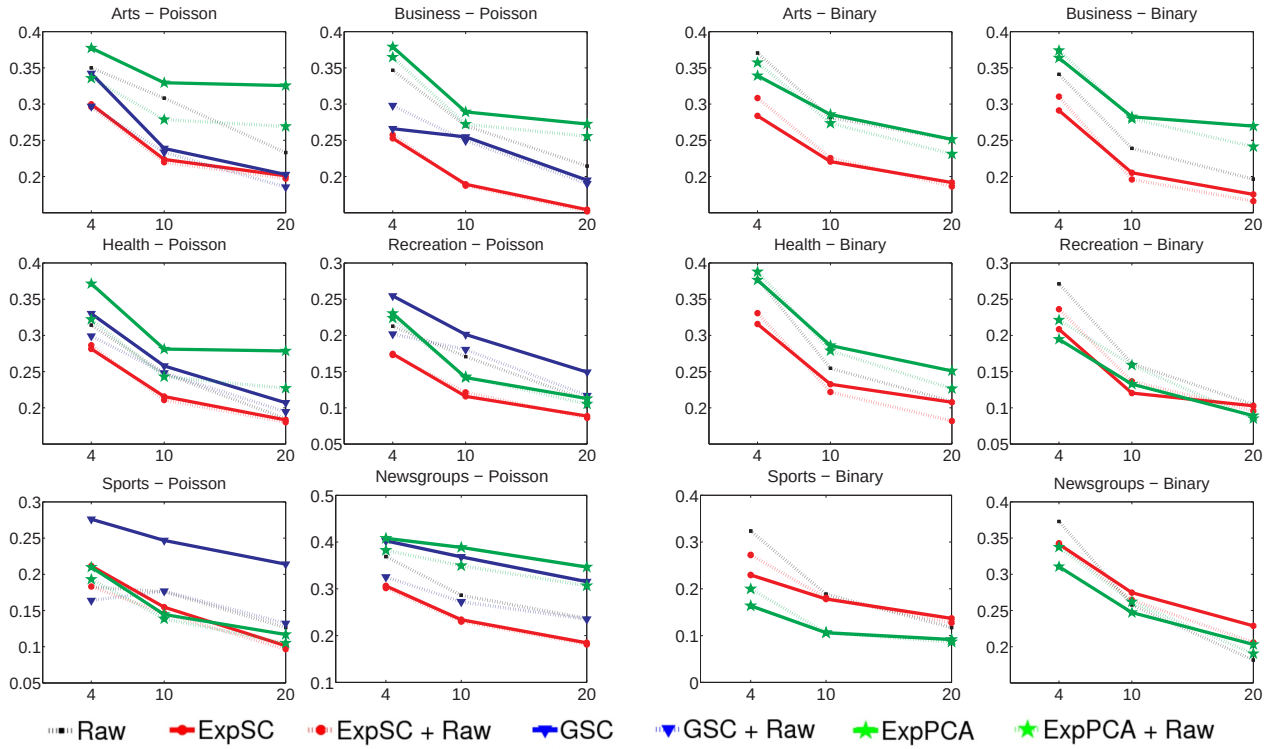

Figure 1: Test error plotted vs. training set size for the webpage and newsgroup classification tasks. Figures in the left half are for Poisson sparse coding, and in the right half for binary sparse coding. See text for explanation of algorithms, and Table 4 for averaged test errors. (Best viewed in color.)

Training set size		Raw	ExpSC	ExpSC +Raw	GSC	GSC +Raw	ExpPCA	ExpPCA +Raw	LDA +Raw
Poisson	4	29.6%	25.4%	**25.0%**	31.2%	26.4%	32.9%	30.4%	33.2%
	10	24.3%	18.9%	**18.6%**	26.1%	22.7%	26.3%	23.8%	24.7%
	20	18.4%	15.2%	**14.9%**	21.4%	17.6%	24.2%	21.2%	17.6%
Binary	4	34.3%	**27.9%**	30.0%	-	-	29.1%	31.3%	-
	10	23.0%	20.5%	**20.4%**	-	-	22.3%	22.7%	-
	20	17.7%	17.4%	**16.1%**	-	-	19.3%	17.7%	-

Table 4: Aggregate test error across all text classification problems (webpages/newsgroups), represented either word count vectors (Poisson) or using binary vectors (Binary). LDA+Raw performed better than LDA alone; so we show results only for this case.

Green, P. J. (1984). Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives. *Journal of the Royal Statistical Society, Series B, Methodological*, *46*, 149–192.

Koh, K., Kim, S.-J., & Boyd, S. (2007). An interior-point method for large-scale L_1-regularized logistic regression. *JMLR*, *8*, 1519–1555.

Lee, H., Battle, A., Raina, R., & Ng, A. Y. (2007). Efficient sparse coding algorithms. *NIPS*.

Lee, S.-I., Lee, H., Abbeel, P., & Ng, A. Y. (2006). Efficient L_1 regularized logistic regression. *AAAI*.

Lokhorst, J. (1999). *The LASSO and Generalised Linear Models*. Honors Project, Department of Statistics, The University of Adelaide, South Australia, Australia.

McCullagh, P., & Nelder, J. A. (1989). *Generalized linear models*. Chapman & Hall.

Ng, A. Y. (2004). Feature selection, L_1 vs. L_2 regularization, and rotational invariance. *ICML*.

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, *381*, 607–609.

Perkins, S., & Theiler, J. (2003). Online feature selection using grafting. *ICML*.

Raina, R., Battle, A., Lee, H., Packer, B., & Ng, A. Y. (2007). Self-taught learning: Transfer learning from unlabeled data. *ICML*.

Roth, V. (2004). The generalized lasso. *IEEE Trans. Neural Networks*, *15*, 16–28.

Weston, J., Collobert, R., Sinz, F., Bottou, L., & Vapnik, V. (2006). Inference with the universum. *ICML*.

Yu, J., Vishwanathan, S. V. N., Günter, S., & Schraudolph, N. N. (2008). A quasi-Newton approach to nonsmooth convex optimization. *ICML*.

Using Prior Domain Knowledge to Build Robust HMM-Based Semantic Tagger Trained on Completely Unannotated Data

Kinfe Tadesse Mengistu KINFE.TADESSE@OVGU.DE
Mirko Hannemann MIRKO.HANNEMANN@GMAIL.COM
Tobias Baum TOBIAS_BAUM@GMX.NET
Andreas Wendemuth ANDREAS.WENDEMUTH@OVGU.DE
Cognitive Systems Group, FEIT-IESK, Otto-von-Guericke University, 39106 Magdeburg, Germany

Abstract

In this paper, we propose a robust statistical semantic tagging model trained on completely unannotated data. The approach relies mainly on prior domain knowledge to counterbalance the lack of semantically annotated treebank data. The proposed method encodes longer contextual information by grouping strongly related semantic concepts together into cohesive units. The method is based on hidden Markov model (HMM) and offers high ambiguity resolution power, outputs semantically rich information, and requires relatively low human effort. The approach yields high-performance models that are evaluated on two different corpora in two application domains in English and German.

1. Introduction

A spoken dialog system with an ideal speech recognizer can barely serve any purpose without a spoken language understanding unit (SLU) that can infer the intention underlying a recognized utterance. Spoken language understanding can be easy for simple application domains where users are restricted in the choice of their formulation of a spoken request. However, the task gets more challenging when a dialog system allows human-to-human like conversation because the natural phenomena of spontaneous speech such as hesitations, false starts, filled pauses, etc. introduce undesirable noise into the input.

Spoken language understanding has been a topic of research since the 70s (Woods, 1983) and spontaneous spoken language understanding has been of particular interest since the early 90s when multiple research laboratories from both academia and industry participated in the DARPA-funded Air Travel Information System (ATIS) evaluations (Price, 1990). In general, the approaches in the domain of spoken language understanding can be grouped as data-driven, rule-based and a combination of the two. Data-driven approaches such as those implemented in CHRONUS of AT&T (Pieraccini & Levin, 1993), and Hidden Understanding Model of BBN (Miller et al., 1994) estimate model parameters from data by counting the frequencies of transitions between states, word observations while in each state and which states start a sentence. These statistical models are robust and perform well but require a large corpus of fully annotated training examples, which is often not practically available. Another popular statistical approach that uses HMMs in SLU is the Hidden Vector State model of Cambridge University (He & Young, 2005). In the Hidden Vector State Model, state transitions between two states are decomposed into separate stack operations that transform one state to the other. A remarkable feature of the HVS model is that it can be trained on "lightly" annotated data and it captures hierarchical structure. Rule-based systems, on the other hand, such as those implemented in TINA of MIT (Seneff, 1992), PHOENIX of CMU (Ward & Issar, 1994), and GEMINI of SRI (Dowding et al., 1994) use handcrafted semantic rules to extract meaning from a spoken utterance. Rule-based systems do not require a large amount of semantically annotated data and they perform very well when the structure of the spoken utterance is covered by the grammar. However, rule-based systems, in general, are very expensive to build and maintain since they require extensive manual in-

Appearing in *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008.

volvement and expertise.

Different combinations of rule-based and statistical approaches have also been investigated. For instance, the generative HMM/CFG (context free grammar) model described in (Wang et al., 2005) integrates a knowledge-based approach into a statistical learning framework.

In this paper, we describe an approach towards spoken language understanding that makes extensive use of a priori domain knowledge in order to build domain-dependent semantic models with relatively less human intervention on completely unannotated data. We essentially add semantic information to the output of the speech recognizer of our dialog system so that a deterministic program can easily infer the intention of the user from the output of the semantic tagger. Assuming that an utterance consists of a sequence of concepts, the purpose of the required model is to determine the most likely sequence of semantic concepts that could have generated the observed sequence of words. The notably distinguishing ability of hidden Markov models (HMMs) to estimate the probability of hidden events from observed ones makes them a natural choice for this kind of task.

The remaining part of the paper is organized as follows. Section 2 briefly describes the architecture of our telephone-based spoken dialog system. The modeling approach is described in detail in Section 3. Section 4 describes the data used in the experiments that are described in Section 5. Finally, concluding remarks are presented in Section 6.

2. Architecture of the Dialog System

As can be seen in the high-level architecture of the system in Figure 1, our VoiceXML-based telephone dialog system consists of a telephony interface component to deliver calls into the system; an input component to accept, recognize and understand spoken requests from a caller; an output component to play prompts and responses back to the user; a back-end to serve dialog scripts and other resources; and at the core is a dialog manager that orchestrates the various components of the system.

The input component of the dialog system consists of an audio source component, a speech recognizer, a grammar (language model) component and a semantic analyzer. The recognition resources used by the recognizer; namely, the acoustic model, the language model, and the pronunciation lexicon are prepared offline using HTK (Young et al., 2006) and the real-time speech recognizer is built using ATK (Young, 2007).

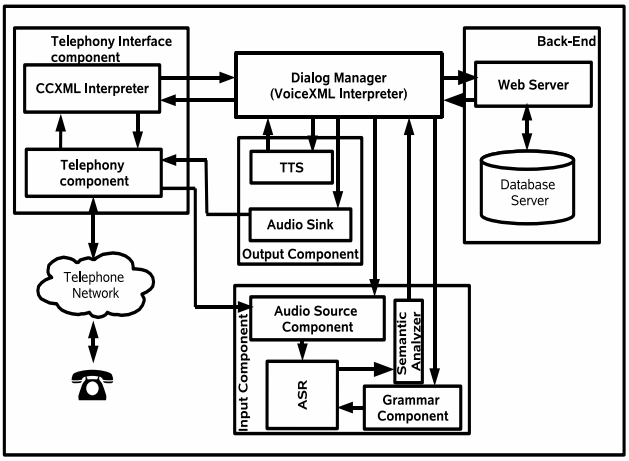

Figure 1. High-level Block Diagram of the Dialog System.

The output of the speech recognizer is sent to the semantic analyzer which we describe in this paper so that the text output of the recognized utterance is enriched with semantic information.

At the core of the system is ©OptimTalk [1] - a VoiceXML platform which consists of not only a VoiceXML interpreter but also a CCXML interpreter, and other abstract interfaces for the integration of our ASR engine, TTS engine, telephony interface, semantic interpreter, etc. The VoiceXML interpreter in OptimTalk serves as the dialog manager in that it executes the dialog by calling the appropriate methods of the various components of the dialog system as shown in Figure 1.

3. Modeling Approach

Understanding an application domain requires a precise identification of the activities, entities, events, attributes and relations within the domain of discourse. The ontologies of the two application domains of interest in this paper - namely, airline travel planning and train inquiries, are modeled in two stages. In the first stage, a detailed list of concepts that are relevant in each application domain are identified using prior domain knowledge and domain-specific example sentences from the training data. As a result, 68 semantic concepts in the domain of airline travel planning and 50 in that of train inquiries are identified. In the second stage, groups of attributes that describe a single semantic concept are grouped together to form cohesive units referred to as super-concepts. For instance, a super-concept DATE contains attributes such as

[1] http://www.optimsys.eu

DAY_OF_MONTH, DAY_OF_WEEK, MONTH, and YEAR. As can be imagined, the prior knowledge used to determine which attributes should belong together to form a super-concept is a commonplace knowledge. Moreover, in order to model multi-word city names and train stations such as "New York City" or "Berlin Friedrichstrasse", we model each with multiple states. The number of sub-concepts in a super-concept can vary; on average, a super-concept contains 3.6 sub-concepts in each application domain.

Accordingly, 14 super-concepts for airline travel planning and 9 for the domain of train inquires are identified. Figure 2 depicts examples of super-concepts along with their attributes (sub-concepts) in the domain of airline travel planning.

CITY (CITY_P1, CITY_P2, CITY_P3, SPELT_CITY),
DATE (DAY_OF_MONTH, DAY_OF_WEEK, MONTH, YEAR),
TIME (MINUTES, HOUR_OF_DAY, AMPM),
AIRLINE (AIRLINE_QUALIFIER, AIRLINE_NAME),
AIRPORT (AIRPORT_NAME, AIRPORT_TYPE,
AIRPORT_QUALIFIER, SPELT_AIRPORT),
CAR_INFO (RENTAL_COMPANY, CAR, CAR_TYPE),
FLIGHT_INFO (FLIGHT_CLASS, FLIGHT_NUMBER,
FLIGHT_TYPE, FLIGHT_QUALIFIER),
HOTEL_INFO (HOTEL_TYPE, HOTEL_QUALIFIER,
LOCATION),
USER (ID, ID_NUMBER, NAME_OF_USER),
PRICE (FARE, AMOUNT_OF_MONEY, FARE_CLASS),

Figure 2. Example super-concepts

Other single state concepts include COUNTRY, STATE, TO, FROM, AT, IN, ON, ARRIVAL, DEPARTURE, RETURN, COMMAND, YES, NO, DUMMY, ...

The rationale behind grouping related sub-concepts together is threefold. First, it improves the predictive power of the model since adjacent related concepts are well coupled. Second, the models produce outputs that are semantically rich and more informative since a phrase is more meaningful than a single word. For instance, phrases like "Saturday the sixteenth of August two thousand eight" or multiword location names such as "Washington D. C." etc. would be more informative if the phrases are labeled as "DATE" and "CITY", rather than tagging each piece with an atomic semantic label. Third, it offers high ambiguity resolution power. For instance, "twenty six" in "November twenty six" would not be confused with other semantic labels such as HOUR, MINUTES, QUANTITY, FLIGHT_NUMBER, ID_NUMBER, etc. as DATE is a super-concept whose attributes are well coupled.

The initial HMMs are defined to be fully connected networks such that any state or sub-network can follow any other single state concept or sub-network with equal probability. Self-loops are allowed in most of the semantic concepts to account for multiple observations from some concepts such as DUMMY, PERIOD_OF_DAY, DAY_OF_MONTH, MINUTES, etc. Every sub-network has its own non-emitting entry and exit states and the transitions between the states within a sub-network are initially ergodic. A one-step transition from the entry state to the exit state of a sub-network is explicitly prohibited to prevent non-emitting loops. Finally, two more non-emitting states INIT and FINAL are introduced to mark the beginning and end of the entire network. Figure 3 shows the partial structure of the HMM for the domain of airline travel planning.

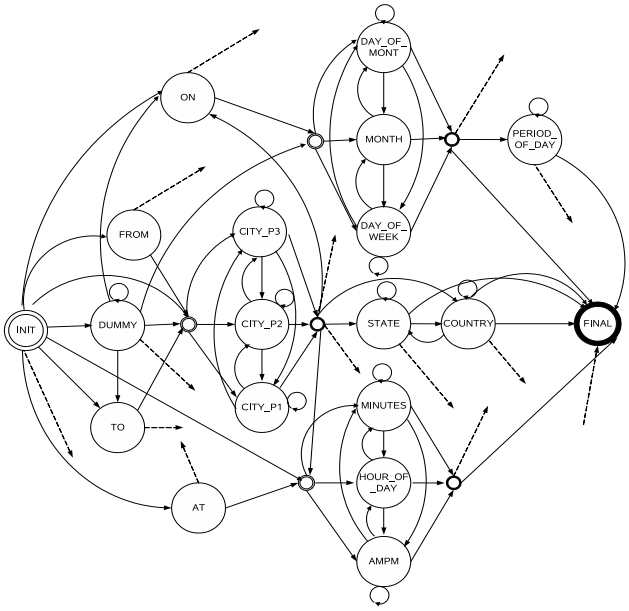

Figure 3. Structure of the HMM

The emission probabilities are initialized by classifying the words in the vocabulary of the application domains into the known set of lexical classes. All words belonging to a semantic label are set equiprobable.

Once we define the model, it may be necessary to bias the transition and emission probabilities of the HMM a bit since we do not have hand-labeled corpus (Elworthy, 1994). This can be done by performing preliminary tests on the training or development data and introducing necessary contraints. For instance, in order to disambiguate words belonging to multiple semantic classes, some unlikely transitions that could not be resolved by the model can be explictly prohibited.

To provide easy tuning and to keep the level of human effort to the minimum, we implemented a model compiler that generates the transition and emission probabilities of the model given the structure and constraints of the HMM in the form shown in Figure 4.

```
INIT
-> except{FINAL}
...
CITY
{
        CITY
        ->except{~CITY}
        CITY_P1
        ->all high{CITY_P2} low{~CITY}
        "city_p1.txt"
        CITY_P2
        ->all high{CITY_P3,~CITY} low{CITY_P1}
        "city_p2.txt"
        CITY_P3
        ->all high{CITY_P3, ~CITY}
        "city_p3.txt"
        SPELT_CITY
        ->all high{SPELT_CITY}
        "spelt_city.txt}
        ~CITY
        ->none
}
->except{INIT} high{AIRPORT, STATE, COUNTRY}
DATE
{
        DATE
        ->except{~DATE}
        MONTH
        ->all high{DAY_OF_MONTH}
        "month.txt"
        DAY_OF_MONTH
        ...
        ~DATE
        ->none
}
->except{INIT}
...
TO
->except{INIT} high{CITY,AIRPORT}
"to.txt"
...
FINAL
->none
```

Figure 4. Excerpt from the model definition

The initial model transition probabilities can be easily tuned as required using the keywords "all", "high", "low", "except", and "none". The model compiler also allows us to try different modeling approaches at different levels of hierarchy with relatively less effort than would be required otherwise. The keywords are self-explanatory; for instance, "− >except{...}" means that all transitions out of a state (e.g. INIT) or sub-network (e.g. CITY) are equally likely except the state(s) specified in braces (e.g. FINAL). The keyword "high" sets a higher probability to the specified transitions than to the rest. For every sub-network the entry state is denoted by the name of the super-concept (e.g. CITY) and the exit state by a tilde followed by the name of the super-concept (e.g. `~CITY`).

Given "well-informed" initial models, the Expectation-Maximization (EM) algorithm can be used to estimate reliable model parameters. The algorithm starts with the carefully defined initial HMM described above and iteratively refines the model parameter values. Once we have a well-trained model, the Viterbi algorithm can be used to find the highest probability semantic label sequence which corresponds to the sequence of observed words.

4. Data Description

The semantic model for airline travel planning domain is trained and evaluated on the transcriptions of speech data from the 2001 DARPA Communicator Evaluation telephone speech corpus (Walker et al., 2003). Table 1 describes the training and test sets of the airline travel planning domain used in the experiments described in Section 5.

Table 1. Description of data for airline travel planning domain

Set	# of Utt.	# of Uniq. Words
Training	8000	915
Test	1000	581

The transcriptions of 8000 utterances were selected from the training data that we used to build the acoustic model for our speech recognizer by removing too many occurences of some very short utterances such as "yes", and "no". For testing purposes, 1000 distinct, relatively longer utterance transcriptions are selected from a 5000 utterance test-set. The average number of words per utterance are 4.04 and 8.98 in the training and test sets of the airline travel planning domain, respectively.

For the domain of train inquiries, we use the transcriptions of utterances from ©Erlanger Bahn Anfragen (ERBA) speech corpus in German. Table 2 describes the data used to build and evaluate the German semantic model.

Table 2. Description of data for train inquiries domain

Set	# of Utt.	# of Uniq. Words
Training	8000	921
Test	1000	830

The utterances in the domain of train inquiries are long, well-structured, and grammatically correct utterances. The average number of words per utterance are 12.26 and 11.76 in the training and test sets, respectively.

5. Experiments and Results

The performance of the systems is evaluated using precision, recall and F-measure where precision (P) is the number of correctly labeled concept chunks out of all tagged concepts, recall (R) is the number of correctly identified concepts from the ground truth annotation. F-measure is the harmonic mean of precision and recall defined by Equation 1

$$F = \frac{2PR}{P+R} \tag{1}$$

5.1. The Effect of the Proposed Modeling Method

If we randomly assign to each token one of its possible tags, we achieve an average performance rate of 56.4% in F-measure. This can be considered as the minimum average baseline performance on the airline travel planning domain.

Table 3 summarizes the performance gain mainly due to the modeling approach we described in Section 3. "Flat" in Table 3 refers to a flat, ergodic HMM model, before grouping of related concepts where each state is one of the sub-concepts or single state concepts described in Section 3. "Grouped" in Table 3 refers to a model just after grouping related sub-concepts together. In both cases no tuning is performed.

Table 3. Flat vs. Grouped initial models on Communicator task

Model	P(%)	R(%)	F-Measure(%)
Flat	57.32	66.42	61.53
Grouped	85.67	77.98	81.64

As can be seen in Table 3, the performance was improved by 20.11% absolute in F-measure just after grouping related concepts together using our prior domain knowledge. This suggests the modeling approach we used is quite suitable for this kind of task.

The experiments that follow describe the performance of both airline travel planning and train inquiries systems stage by stage using the proposed modeling approach.

5.2. Performance of the Initial Models after Tuning

Table 4 depicts the performance of the initial models tuned as described in Section 3 before EM training.

Table 4. Performance of the tuned initial models

Data	P(%)	R(%)	F-Measure(%)
Communicator	96.15	84.46	89.92
ERBA	94.90	94.19	94.54

As can be observed, the performance of the initial model after tuning is improved significantly.

5.3. Performance of the Models after Training

The results after performing EM training are summarized in Tables 5.

Table 5. Performance of the models after training

Data	P(%)	R(%)	F-Measure(%)
Communicator	98.75	84.58	91.12
ERBA	96.94	96.19	96.56

The best performance was achieved after only one iteration of training on the airline travel planning and two on the train information inquiries domain, respectively. It can be noted in Table 5 that the recall, mainly for the airline travel planning domain, is quite low. This is due to unseen transitions and "out-of-vocabulary" words (OOVs) that resulted in some unparseable utterances. This is, in turn, attributed to the sparse data problem and the inevitability of OOVs. In order to combat this problem, we smoothed transition and emission probabilities. The recall in the train inquiries domain is high because the rate of OOVs in the German test-set is low.

5.4. Performance after Smoothing

In the case of transitions, we assigned a very small nonzero probability for all allowable transitions not seen in the training data. For words not found in the lexicon, we introduced a vocabulary item "oov" in those lexical classes where there is no exhaustive list of words; for instance, CITY, DUMMY, AIRLINE, etc. The probability of the "oov" word in a concept is set to the sum of the probabilities of all words belonging to that concept that occur only once in the training set; the rest of the probabilities are discounted accordingly so

that all sum up to one. As a result, all the sentences could be parsed and 69% of the "oov" words out of 135 in the domain of airline travel planning were correctly labeled.

Accordingly, the performance of the models is significantly improved as can be seen in Table 6.

Table 6. Performance of the models after smoothing

DATA	P(%)	R(%)	F-MEASURE(%)
COMMUNICATOR	96.91	97.12	97.01
ERBA	96.82	97.41	97.11

5.5. Example Tagged Outputs

An example tagged output of an utterance in the domain of airline planning is:
(I want to fly) DUMMY (from) FROM (Los Angeles) CITY (to) TO (Osaka) CITY (Japan) COUNTRY (on) ON (September thirtieth) DATE (in the morning) PERIOD_OF_DAY
An example tagged output of an utterance in the domain of train inquiries in German[2] is:
(ich moechte) DUMMY (alle) MODIFIER (Abfahrts) DEPARTURE (und) CONNECTIVE (Ankunftszeiten) ARRIVAL (aller) MODIFIER (Zuege) TRAIN (nach) TO (Minden) LOCATION (an) ON (einem) DUMMY (Wochentag) DATE (zwischen neun und sechs Uhr) PERIOD_OF_DAY

6. Conclusions

In this paper, we described the use of prior domain knowledge to build an HMM-based semantic tagging model with four main virtues - namely, it is trained on completely unlabeled data, it offers high ambiguity resolution ability, it outputs naturally appealing and semantically rich information, and it requires relatively low human effort as the model itself takes care of many sources of ambiguity. Moreover, the model is robust in that it could parse unseen observations and could correctly label a significant amount of out-of-vocabulary words. The performance of the proposed model is 97.01% for airline travel planning and 97.11% for train-inquires system in F-measure on 1000-utterance test-sets. The success of this approach relies mainly on the use of a priori domain knowledge to build a reliable initial model while keeping the human effort to the minimum.

[2]Translation of the German utterance: I want all departure and arrival times of all trains to Minden on a weekday between nine and six o'clock

References

Dowding, J., Moore, R., Andry, F., & Moran, D. (1994). Interleaving syntax and semantics in an efficient bottom-up parser. *Proc. 32nd Annual Meeting of the Association for Computational Linguistics*, 110–116.

Elworthy, D. (1994). Does Baum-Welch Re-estimation Help Taggers? *4th Conference on Applied Natural Language Processing, pages 53-58. Association for Computational Linguistics*, 53–58.

He, Y., & Young, S. (2005). Semantic Processing using the Hidden Vector State Model. *Proceedings of Computer Speech and Language*, *19*, 85–106.

Miller, S., Bobrow, R., Ingria, R., & Schwartz, R. (1994). Hidden understanding models of natural language. *Proceedings of the Association of Computational Linguistics*, 25–32.

Pieraccini, R., & Levin, E. (1993). A learning approach to natural language understanding. *NATO-Advanced Study Institute: New Advances & Trends in Speech Recognition and Coding*, *1*, 261–279.

Price, P. (1990). Evaluation of Spoken Language System: The ATIS Domain. *DARPA Speech and Natural Language Workshop.*

Seneff, S. (1992). TINA: A natural language system for spoken language applications. *Comput. Linguistics*, *18*, 61–86.

Walker, M., Aberdeen, J., & Sanders, G. (2003). 2001 communicator evaluation.

Wang, Y.-Y., Deng, L., & Acero, A. (2005). Spoken language understanding: An introduction to statistical framework. *IEEE Signal Processing Magazine*, *19*, 85–106.

Ward, W., & Issar, S. (1994). Recent improvements in the CMU spoken language understanding system. *Proceedings of the ARPA Human Language Technology Workshop*, 213–216.

Woods, W. A. (1983). *Language Processing for Speech Understanding.* Englewood Cliffs, NJ: Computer Speech Processing, Prentice-Hall International.

Young, S. (2007). ATK:An Application Toolkit for HTK Version 1.6.1.

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X. A., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., & Woodland, P. (2006). The HTK Book. Revised for HTK Version 3.4.

Knowledge as a Constraint on Uncertainty for Unsupervised Classification: A Study in Part-of-Speech Tagging

Thomas J. Murray IV TMURRAY@USC.EDU
University of Southern California, Speech Analysis and Interpretation Laboratory, Los Angeles, CA 90089 USA

Panayiotis G. Georgiou GEORGIOU@SIPI.USC.EDU
University of Southern California, Speech Analysis and Interpretation Laboratory, Los Angeles, CA 90089 USA

Shrikanth S. Narayanan SHRI@SIPI.USC.EDU
University of Southern California, Speech Analysis and Interpretation Laboratory, Los Angeles, CA 90089 USA

Abstract

This paper evaluates the use of prior knowledge to limit or bias the choices of a classifier during otherwise unsupervised training and classification. Focusing on effects in the uncertainty of the model's decisions, we quantify the contributions of the knowledge source as a reduction in the conditional entropy of the label distribution given the input corpus. Allowing us to compare different sets of knowledge without annotated data, we find that label entropy is highly predictive of final performance for a standard Hidden Markov Model (HMM) on the task of part-of-speech tagging. Our results show too that even basic levels of knowledge, integrated as labeling constraints, have considerable effect on classification accuracy, in addition to more stable and efficient training convergence. Finally, for cases where the model's internal classes need to be interpreted and mapped to a desired label set, we find that, for constrained models, the requirements for annotated data to make quality assignments are greatly reduced.

1. Introduction

This paper investigates one of the simplest methods for integrating prior knowledge into the training of an unsupervised classifier, in particular the restriction or, more generally, weighting of output labels for each given input. In this setting, we focus on how the knowledge source constrains the set of available choices for the learner, effectively reducing the uncertainty in the classification decision. More precisely, viewing this guidance as a distribution over label output for the input data, we then may quantify and compare the effects of different sets of knowledge in terms of conditional entropy, without the need for annotated data.

To evaluate the relationship between knowledge constraints, uncertainty, and classification performance, we take the basic task of part-of-speech tagging, with the standard, first-order Hidden Markov Model (HMM) tagger of Merialdo (1994). We apply a number of basic constraint sets during training and evaluation, from lexical rules to partial tagging dictionaries, and find that the conditional label entropy is highly predictive of final model performance, with even the weakest constraints leading to large increases in classification accuracy. In addition, we see considerable reductions of variance in performance with respect to initial conditions and accelerated training convergence. Finally, addressing the problem of assigning interpretable labels to internal model classes, we find that the more constrained models require much less annotated data to find quality mappings.

The remainder of the paper is organized as follows. After discussing related work in the next section, we formalize the learning setting and clarify our entropy calculation in Section 3. Following a brief description of our constraint sets in Section 4, we present our results in Section 5 and conclude with a discussion in Section 6.

Appearing in *Workshop on Prior Knowledge for Text and Language*, Helsinki, Finland, 2008.

2. Related Work

A major impetus of the present study was Johnson's (2007) comparison of Expectation-Maximization (EM) and Bayesian estimators for unsupervised tagging, in particular the following conclusions of that work: (1) EM can be competitive with more sophisticated Bayesian methods, (2) greatly subject to the choice of evaluation method, but (3) to a certain extent, it is possible to compensate for EM's weakness in estimating skewed distributions by constraining the model to exclude rare events. While this is certainly not an argument against the use of better estimators, it does suggest the potential benefits of even simple means to guide model training.

Merialdo (1994) introduced the statistical tagging models employed in this paper and many, many others, both for supervised and unsupervised training. Like the original, much subsequent work on unsupervised tagging has relied on a full dictionary of possible tags for each word, which in practice constrains the training sufficiently to obviate the need for remapping of model output. More recent approaches with discriminative models (Smith & Eisner, 2005) and Bayesian estimators (Goldwater & Griffiths, 2007) have achieved good performance while reducing the dictionary. Haghighi and Klein (2006) require only a limited set of class prototypes and representations of their context distributions, while Toutanova and Johnson (2008) use only the possible groups of tags over which words are seen to vary.

Our simple integration of knowledge constraints may be viewed as an instance of virtual evidence (Pearl, 1988), a method to account for external judgements not easily expressed in terms of the probability distributions and dependencies encoded by the model. Bilmes (2004) suggests virtual evidence as a means to integrate the assessments of external models. Chang et al. (2007) uses weighted constraints successfully to guide search in an iterative algorithm for semi-supervised learning.

3. Knowledge as a Constraint on Uncertainty

To evaluate the effects of prior knowledge in entropic terms, we extend the usual formulation of a classifier to include, along with each input x, a mapping $\phi_x(y)$ of each output label y to a non-negative weight. That is, we define a classifier as a mapping $\mathcal{X} \times (\mathcal{Y} \to \mathbb{R}) \to \mathcal{Y}$. In the supervised case, $\phi_x(y)$ is non-zero for exactly one value of y, while in the purely unsupervised case, the weights are equal for all values of y. Normalizing the weights and interpreting them as a distribution $p(Y = y|X = x)$, the conditional entropy $H(Y|X)$ is a natural measure of the uncertainty facing the classifier and a means to compare and predict the effects of different sets of knowledge.[1] Because the uniform distribution has the maximum entropy for an event space of a given cardinality (Cover & Thomas, 1991), any adjustment to the label weights must reduce entropy relative to the purely unsupervised case. Assuming this reweighting does not penalize or eliminate the correct label, we expect a similar improvement to model accuracy.

3.1. Entropy Calculation

For sequential labeling tasks such as part-of-speech tagging, we generally label each token individually to avoid sparsity in our estimation, and so it is natural to take the x and y in the $p(y|x)$ above to refer to a single input token and label. Some of the constraints in our experiments involve context, however, so that $\phi_x(y)$ may vary between instances of x in the corpus. Accordingly, we must introduce some notion of context C and calculate the entropy as

$$\begin{aligned} H(Y|X,C) &= \sum_x \sum_c p(x,c)\, H(Y|X=x, C=c) \\ &= \sum_x p(x) \sum_c p(c|x) \sum_y p(y|x,c) \log p(y|x,c) \end{aligned}$$

If, however, we estimate $p(c|x)$ by simple counts over contexts that are equivalent under our constraints, we effectively sum out c, computing $H(Y|X = x)$ as an average of $H(Y|X = x, C)$ in all contexts. Thus, while our calculations use $p(y|x, c)$, we will continue to speak of $H(Y|X)$ for the remainder of the paper.

4. Constraints on Unsupervised Tagging

Great care is always required to evaluate an unsupervised classifier fairly against a labeled corpus, but evaluation is even more of a delicate matter when additional prior knowledge is involved. Ideally our knowledge should not be derived from the corpus, or we are crossing the line into supervised learning, but, practically, a successful set of constraints must accord with the knowledge implicit in the annotated data and be expressed in similar terms.[2] Accordingly, unless noted otherwise, we construct the following constraint sets

[1] This value also accords with the common informal characterization of classification difficulty in terms of average possible labels per input element.

[2] For example, an educated speaker of English would differentiate between the uses of the word 'to' as a preposition

from the knowledge of a native speaker and from general grammar resources, independent of the corpus.

Base Lexical Constraints For our base rule set, we include knowledge about punctuation, numbers, and capitalization, with numbers forced to either contain a digit or recursively to follow another possible number (to handle, e.g. '10 million'), and proper nouns defined similarly with respect to capitalization.

Closed Tags For each closed part-of-speech tag, we then add (incomplete) lists of possible words, derived from external sources as available.

Top Words Like much work on unsupervised tagging, we finally apply a tagging dictionary built from the corpus, but only for the 100 or 200 most common words, a much more limited amount of annotation.

For most of the experiments, we apply the above as hard constraints, with all violating hypotheses excluded, but we also examine the use of soft constraints, where hypotheses that observe the rules are simply preferred.

5. Experimental Results

5.1. Experimental Overview

In our experiments, we performed unsupervised training of the simple first-order HMM tagging model of Merialdo (1994), using the EM algorithm with a variety of constraint sets. Our implementation used the Graphical Models Toolkit (GMTK) (Bilmes & Zweig, 2002), with hard and soft constraints integrated via the toolkit's deterministic node construct.[3] For training and evaluation, we used the Wall St. Journal portion of the Penn Treebank, version 3 (Marcus et al., 1993), with data sets containing the 48k, 96k, and 193k words following the start of section 2.

To account for the local search properties of EM, we repeated each experiment 10 times, training for 500 iterations, with parameters initialized by small, random perturbations from the uniform distribution. Because our constraints cause no changes to the model's parameter set, it was possible to use the same random initializations across constraints for each data set, and thus attempt to control for any bias from particularly good or bad initialization points.[4]

For evaluation, we used the 'many-to-one' and 'one-to-one' labeling procedures as described by Johnson (2007), which greedily assign each model state to the annotated tag with which it occurs most often, respectively either allowing or prohibiting multiple states to map to a single tag. While, as Johnson (2007) and (Haghighi & Klein, 2006) mention, we may cheat with the many-to-one labeling by inflating the number of model states, this flaw seems less critical if the state count equals the size of the tag set, as in our experiments.

5.2. Results and Analysis

As summarized in Table 1 we find that even the least constrained models show considerable improvement over the baseline, with up to 20-30 percentage points gained in accuracy. Despite the simplistic nature of the model, performance is often surprisingly close to much more sophisticated models and training techniques, e.g. (Smith & Eisner, 2005; Haghighi & Klein, 2006; Goldwater & Griffiths, 2007). As we might expect, the effects are most pronounced on the smaller data sets, where the constraints serve as a strong prior compensating for lack of evidence, similar to what we see with the Bayesian models of Goldwater and Griffiths (2007). The effect on both the many-to-one and one-to-one label assignments is roughly equal across experiments, so that the difference in accuracy between the two assignments changes little as we add constraints.

To assess the effect of uncertainty on final model performance, we computed the Pearson correlation coefficient r^2 between the label entropy and the classifier accuracy, as shown in Figure 1. While the two are not fully correlated – and we should not necessarily expect them to be – the entropy measure is quite indicative of performance, and we conclude that is a reasonable means for predicting the effects of domain knowledge when annotated data is unavailable for evaluation.

Of course, knowledge is helpful only if the correct answer is not among the excluded hypotheses. We explored the effects of imperfect knowledge by applying our closed-tag rules set as a hard constraint and then as a soft constraint with relative likelihood weights ranging from 2:1 to 16:1. Though these rules are incomplete for most of the tags covered, and thus the correct labels for many words were excluded, we saw the hard constraints perform best, edging out the highest-weight soft constraints. It appears that, while

and as an infinitive verb marker, but in the Penn Treebank corpus, both are labeled 'TO'.

[3] We thank Chris Bartels for assistance on model implementation.

[4] From casual inspection, however, we did not see any performance patterns across runs. One might argue that different constraints lead to completely different optimization surfaces and extrema under EM.

Model	48k			96k			193k		
	$H(Y\|X)$	N:1	1:1	$H(Y\|X)$	N:1	1:1	$H(Y\|X)$	N:1	1:1
Base	5.49	33.8 (3.7)	21.7 (2.8)	5.49	42.9 (4.4)	30.1 (3.2)	5.49	52.1 (2.5)	34.4 (3.1)
Lower case	5.49	42.3 (2.2)	29.7 (2.3)	5.49	48.9 (2.4)	34.6 (2.5)	5.49	52.7 (2.3)	36.8 (1.9)
+Baselex	4.31	53.6 (0.8)	39.8 (1.9)	4.29	57.3 (0.8)	42.4 (1.6)	4.30	60.7 (0.8)	43.9 (1.7)
+Closed	3.71	64.9 (0.8)	54.3 (0.8)	3.69	66.2 (0.5)	55.5 (0.9)	3.70	67.4 (0.6)	56.4 (0.6)
+Top 100	3.49	69.2 (0.0)	57.8 (0.3)	3.47	70.1 (0.1)	58.6 (0.2)	3.48	71.0 (0.2)	59.5 (0.1)
+Top 200	3.49	**71.9** (0.1)	**60.5** (0.6)	3.47	**72.8** (0.1)	**61.7** (0.3)	3.48	**73.8** (0.1)	**62.1** (0.3)

Table 1. Tagging accuracy with increasing knowledge (as measured by conditional label entropy) on different data sets. Models were evaluated using the many-to-one and one-to-one label assignments, with results averaged over 10 runs, standard deviation in parentheses. Except for the base model, all words were mapped to lower case to reduce vocabulary size.

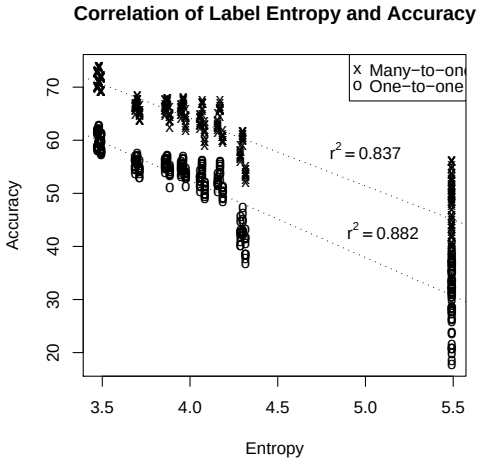

Figure 1. Correlation between conditional entropy $H(Y|X)$ and accuracy, for all runs on the 193k data set.

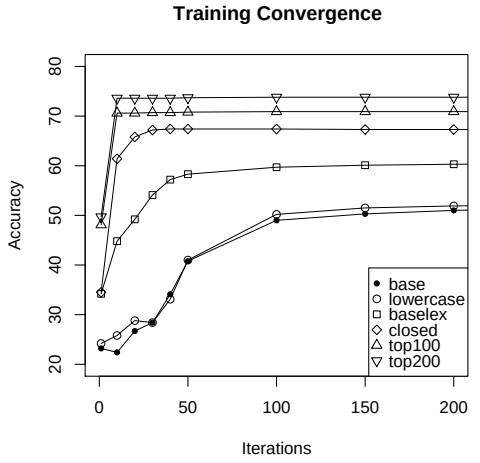

Figure 2. Mean accuracy for constraint sets over training iterations (only minor increases after 200), for 45-state bigram, 193k data set.

the hard rules forced errors, the most common words in each tag were covered fairly well by the grammar lists we used, and the extra reduction in uncertainty outweighed the more obscure errors. A similar effect is observed in Banko and Moore (2004), where they find quite significant gains by filtering out the rare tags of each word. This does not mean necessarily that only hard constraints are useful (indeed, Chang et al. (2007) finds soft constraints to be superior), but it seems they can be beneficial even when they oversimplify the facts, especially for a simple model that has little hope of labeling rare and difficult events correctly. We assume, too, that it would be more ideal to separate rules according to our confidence in them, and assign weights accordingly.

Finally we found that increased knowledge constraints lead to a reduction in the variance of model performance across runs, a major benefit given the problems of local extrema in most unsupervised methods and the difficulty of choosing an optimal model without annotated data. For our most constrained 'Top 100' and 'Top 200' model sets, the standard deviation of the accuracy was generally under 0.5 percentage points. Additional knowledge also constrained the training process, with accuracy converging in fewer iterations. Figure 2 plots the accuracy for models trained on the 193k data set, illustrating how the addition of rules leads to a steeper optimization surface for EM.

5.3. Labeling and Annotation

While the use of fully annotated data to label internal model states is a practical necessity of evaluation in this and similar work, such an artificial scenario is problematic for those real-world situations where classifier output needs to be interpreted or passed to another component in the pipeline (i.e. not just treated as clustering). In such cases, we face the question of how much labeled data is required to perform a quality label assignment.

To explore this issue, we labeled the output of different models trained on the 193k data set, but with only

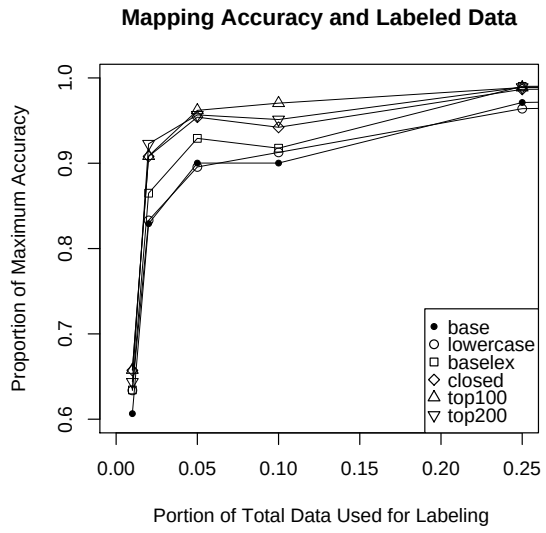

Figure 3. Accuracy convergence of many-to-one labeling methods, as increasing portions of the training data annotations are used to make label assignments, for models trained on the 193k corpus.

part of the annotated data available to generate the mappings. Figure 3 shows the results of the many-to-many method, plotting each data set proportion with the accuracy of the induced label mapping, relative to the best accuracy when all data was used.[5] As an example, consider the case of the unconstrained, lowercased model. With 10% of the data, or roughly 19k words, the labeling accuracy was 48.1, compared to 52.7 when the entire set is used, so that this partial-data assignment performs at 0.91 of its full accuracy.

Our first impression is that labeling performance converges relatively quickly, but we should note that even the 5% portion represents nearly 10,000 words of annotation. Still, with the more constrained knowledge sets, 90% of optimal accuracy is reached with only 2% of the data (4k words), so once again the use of prior knowledge is extremely beneficial in a practical setting.

6. Discussion

We have presented the view of prior knowledge as a reduction of uncertainty in the training of unsupervised classifiers, showing label entropy to be an effective and predictive measure of the contributions of that knowledge, and a means for assessment without annotated data. We found that quite basic domain knowledge can lead to significant performance improvements, with the additional benefits of faster training convergence, better stability, and reduced data requirements for label mapping. While the effects here are no doubt exaggerated by our impoverished models and data sets and the simplicity of the task, our results suggest that even the simple integration of prior knowledge is worthwhile where labeled data is lacking.

[5] One-to-one convergence was slightly faster, but the relative rates of the different constraints were similar.

Acknowledgments

We thank Rahul Bhagat, Abe Kazemzadeh, and the anonymous reviewers for comments on drafts of this paper, and Jorge Silva for fruitful discussion of the work.

This research was supported by the DARPA TRANSTAC program, grant #FA8750-06-1-0250, and by a USC Viterbi School of Engineering Doctoral Fellowship for the first author.

References

Banko, M., & Moore, R. C. (2004). Part of speech tagging in context. *Proc. COLING.*

Bilmes, J. (2004). *On soft evidence in Bayesian networks* (Technical Report UWEETR-2004-00016). University of Washington Dept. of Electrical Engineering.

Bilmes, J., & Zweig, G. (2002). The Graphical Models Toolkit: An open source software system for speech and time-series processing. *Proc. ICASSP.*

Chang, M., Ratinov, L., & Roth, D. (2007). Guiding semi-supervision with constraint-driven learning. *Proc. ACL.*

Cover, T. M., & Thomas, J. A. (1991). *Elements of information theory.* Wiley-Interscience.

Goldwater, S., & Griffiths, T. L. (2007). A fully Bayesian approach to unsupervised part-of-speech tagging. *Proc. ACL.*

Haghighi, A., & Klein, D. (2006). Protoype-driven learning for sequence models. *Proc. HLT-NAACL.*

Johnson, M. (2007). Why doesn't EM find good HMM POS-taggers? *Proc. EMNLP.*

Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics, 19.*

Merialdo, B. (1994). Tagging English text with a probabilistic model. *Computational Linguistics, 20,* 155—171.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann.

Smith, N. A., & Eisner, J. (2005). Contrastive estimation: Training log-linear models on unlabeled data. *Proc. ACL*.

Toutanova, K., & Johnson, M. (2008). A Bayesian LDA-based model for semi-supervised part-of-speech tagging. *Proc. NIPS*.

Dirichlet Process Mixture Models for Verb Clustering

Andreas Vlachos AV308@CL.CAM.AC.UK
Computer Laboratory, University of Cambridge, Cambridge, CB3 0FD, UK

Zoubin Ghahramani ZOUBIN@ENG.CAM.AC.UK
Department of Engineering, University of Cambridge, Cambridge, CB2 1PZ, UK

Anna Korhonen ALK23@CAM.AC.UK
Computer Laboratory, University of Cambridge, Cambridge, CB3 0FD, UK

Keywords: non-parametric modelling, computational linguistics, clustering, semantics, semi-supervised learning

Abstract

In this work we apply Dirichlet Process Mixture Models to a learning task in natural language processing (NLP): lexical-semantic verb clustering. We assess the performance on a dataset based on Levin's (1993) verb classes using the recently introduced V-measure metric. In, we present a method to add human supervision to the model in order to to influence the solution with respect to some prior knowledge. The quantitative evaluation performed highlights the benefits of the chosen method compared to previously used clustering approaches.

1. Introduction

Bayesian non-parametric models have received a lot of attention in the machine learning community. These models have the attractive property that the number of components used to model the data is not fixed in advance but is actually determined by the model and the data. This property is particularly interesting for natural language processing (NLP) where many tasks are aimed at discovering novel, previously unknown information in corpora.

In this work, we apply the basic models of this class, Dirichlet Process Mixture Models (DPMMs) (Neal, 2000) to a typical unsupervised learning task in NLP: lexical-semantic verb clustering. The task involves discovering classes of verbs similar in terms of their syntactic-semantic properties (e.g. MOTION class for the verbs "travel", "walk" and "run"). Such classes can provide important support for other NLP tasks and applications. Although some fixed classifications are available (e.g. Levin (1993)), these are not comprehensive and are inadequate for specific domains such as the biomedical one (Korhonen et al., 2006b).

The clustering algorithms applied to this task so far require the number of clusters as input (Schulte im Walde, 2006; Korhonen et al., 2006b). This is problematic as we do not know how many classes exist in the data. Even if the number of classes in a particular dataset was known (e.g. in the context of a carefully controlled experiment), a particular dataset may not contain instances for all the classes. Moreover, each class is not necessarily contained in one cluster exclusively, since the target classes are defined manually without taking into account the feature representation used. The fact that DPMMs do not require the number of target clusters in advance, renders them particularly promising for the many NLP tasks where clustering is used for learning purposes.

In addition to applying the standard DPMM to verb clustering we also present a method to add human supervision to the model in order to to influence the solution with respect to some prior intuition or some considerations relevant to the application in mind. We achieve this by enforcing pairwise clustering constraints on the solution discovered by the model. We evaluate these methods on two different datasets including general English and biomedical verbs, respectively. Our results compare favourably to earlier results reported with verb clustering and demonstrate the potential of DPMM based models for discovering novel information from natural language data.

Appearing in *ICML Workshop on Prior Knowledge for Text and Language Processing*, Helsinki, Finland, 2008.

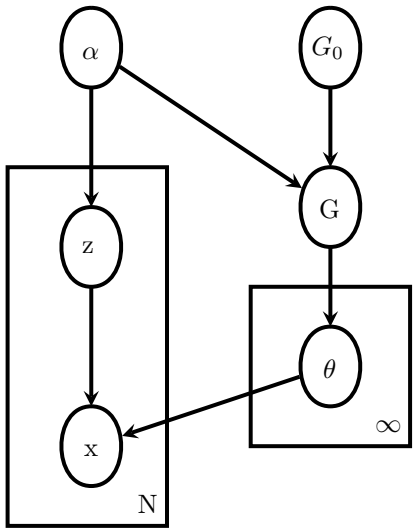

Figure 1. Graphical representation of DPMMs.

2. Unsupervised clustering with DPMMs

With DPMMs, as with other Bayesian non-parametric models, the number of mixture components is not fixed in advance, but is determined by the model and the data. The parameters of each component are generated by a Dirichlet Process (DP) which can be seen as a distribution over the parameters of other distributions. In turn, each instance is generated by the chosen component given the parameters defined in the previous step:

$$\begin{aligned} G|\alpha, G_0 &\sim DP(\alpha, G_0) \\ \theta_{z_i}|G &\sim G \\ x_i|\theta_{z_i} &\sim p(x_i|\theta_{z_i}) \end{aligned} \tag{1}$$

In Eq. 1, G_0 and G are probability distributions over the component parameters (θ), and $\alpha > 0$ is the concentration parameter which determines the variance of the Dirichlet process. We can think of G as a randomly drawn probability distribution with mean G_0. Intuitively, the larger α is, the more similar G will be to G_0. z_i is the component chosen for instance x_i, and θ_{z_i} its parameters. The graphical model is depicted in Figure 1.

The prior for assigning instance x_i to either an existing component z or to a new one z_{new} conditioned on the other component assignments (z_{-i}) is given by:

$$\begin{aligned} p(z_i = z|z_{-i}) &= \frac{n_{-i,z}}{N-1+\alpha} \\ p(z_i = z_{new}|z_{-i}) &= \frac{\alpha}{N-1+\alpha} \end{aligned} \tag{2}$$

where $n_{-i,z}$ is the number of instances assigned to component z excluding instance x_i and N is the total number of instances. A clustering of the instances is generated by assigning more than one instance to the same mixture component.

The prior in Eq. 2 exemplifies two main properties of the DPMMs. Firstly, the probability of assigning an instance to a particular component is proportionate to the number of instances already assigned to it ($n_{-i,z}$). In other words, DPMMs exhibit the "rich get richer" property. Secondly, the probability that a new cluster is created depends on the concentration parameter α.

A popular metaphor to describe DPMMs is the Chinese Restaurant Process. Customers (instances) arrive at a Chinese restaurant which has an infinite number of tables (components). Each customer chooses to sit at one of the tables that is either occupied ($p(z_i = z|z_{-i})$) or vacant ($p(z_i = z_{new}|z_{-i})$). Popular tables attract more customers.

An alternative view of DPMMs is the stick-breaking construction (Sethuraman, 1994). In this construction, the mixing proportions of the components (π_k) are produced as follows:

$$\begin{aligned} \pi_k &= \beta_k \prod_{j=1}^{k-1}(1-\beta_j) \\ \beta_k &\sim \mathcal{B}(1, \alpha) \end{aligned} \tag{3}$$

where $\mathcal{B}$ is the Beta distribution. It can be verified that $\sum_{k=1}^{\infty} \pi_k = 1$. Intuitively, the mixing proportion of each component is obtained by successively breaking a stick of unit length. As a result, the mixing proportion of a new component gets progressively smaller. In order to generate an instance x_i, the component z_i is chosen using a multinomial distribution parameterized by the mixing proportions π_k, and the instance is generated as in Eq. 1.

3. Evaluation

The evaluation of unsupervised clustering against a gold standard is not straightforward because the clusters found by the algorithm are not associated with the classes in the gold standard. Formally defined, the method partitions a set of instances $X = \{x_i|i = 1, ..., N\}$ into a set of clusters $K = \{k_j|j = 1, ..., |K|\}$. To evaluate the quality of the resulting clusters, we use an external gold standard in which the instances are partitioned into a set of classes $C = \{c_l|l = 1, ..., |C|\}$. The aim of a clustering algorithm is to find a partitioning of the instances K that resembles as closely as possible the gold standard C.

Most work on verb clustering has used F-measure or the Rand Index (Rand, 1971) for quantitative eval-

uation. However, Rosenberg and Hirschberg (2007) point out that F-measure assumes (the missing) mapping between c_l and k_j. Also, in their experimental assessment they show that when the number of clusters not representing a particular class was increased the Rand Index did not decrease. Another recently introduced metric, variation information (Meilă, 2007), while it avoids these problems, its value range depends on the maximum number of classes $|C|$ and clusters $|K|$ involved in the evaluation, rendering the performance comparisons between different algorithms and/or datasets difficult (Rosenberg & Hirschberg, 2007). Rosenberg and Hirschberg suggest a new information-theoretic metric for clustering evaluation: V-measure. V-measure is the harmonic mean of homogeneity and completeness which evaluate the quality of the clustering in a complementary way. Homogeneity assesses the degree to which each cluster contains instances from a single class of C. This is computed as the conditional entropy of the class distribution of the gold standard given the clustering discovered by the algorithm, $H(C|K)$, normalized by the entropy of the class distribution in the gold standard, $H(C)$. Completeness assesses the degree to which each class is contained in a single cluster. This is computed as the conditional entropy of the cluster distribution discovered by the algorithm given the class, $H(K|C)$, normalized by the entropy of the cluster distribution, $H(K)$. In both cases, we subtract the resulting ratios from 1 to associate higher scores with better solutions:

$$h = 1 - \frac{H(C|K)}{H(C)}$$
$$c = 1 - \frac{H(K|C)}{H(K)}$$
$$V = \frac{2 * h * c}{h + c} \quad (4)$$

We should note that V-measure favors clustering solutions with a large number of clusters (large $|K|$), since such solutions can achieve very high homogeneity while maintaining reasonable completeness (Rosenberg & Hirschberg, 2007). To demonstrate this bias for the dataset used in the following section, the clustering solution in which each verb is assigned to a singleton cluster achieves 100% homogeneity, 53.3% completeness and 69.5% V-measure, which are in fact higher than the scores achieved by any of the clustering methods evaluated in the following sections. While increasing $|K|$ does not guarantee an increase in V-measure (splitting homogeneous clusters would reduce completeness without improving homogeneity), it is easier to achieve higher scores when more clusters are produced. The lenience of V-measure towards such solutions reflects the intuition mentioned in the introduction that a single class is likely to be contained in more than one cluster given the representation used. As our method does not require the number of clusters in advance, it is worth keeping this bias in mind.

4. Experiments

Following Kurihara et al. (2007), we used variational inference in order to perform parameter estimation for the DPMMs. In particular, we approximated the infinite vector of the mixing proportions using a finite symmetric Dirichlet prior. The distributions generating the instances of each component were Gaussians with diagonal covariance. The initial number of components was set to 100 and the concentration parameter alpha was set to 1.[1]

After inferring the parameters of the DPMM from the data, for each instance we obtain a probability distribution over the components, in other words a "soft" clustering. In order to produce a clustering solution in which each instance is assigned to one cluster only, each instance is assigned to the component with the highest probability. As a result, the components of the mixture are considered to be the clusters of the clustering solution. However, the transformation described above can result in fewer clusters than components, since there may be components that are not the most probable ones for any instance of the dataset, resulting in empty clusters.

To perform lexical-semantic verb clustering we used the dataset of Sun et al. (2008). It contains 204 verbs belonging to 17 fine-grained classes in Levin's (1993) taxonomy so that each class contains 12 verbs. The classes and their verbs were selected randomly. In Sun et al.'s dataset, the features for each verb are their subcategorization frames (SCFs) and associated frequencies in corpus data, which capture the syntactic context in which the verbs occur in text. SCFs were extracted from the publicly available VALEX lexicon (Korhonen et al., 2006a). VALEX was acquired automatically using a domain-independent statistical parsing toolkit, RASP (Briscoe & Carroll, 2002), and a classifier which identifies verbal SCFs. As a consequence, it includes some noise due to standard text processing and parsing errors and due to the subtlety of argument-adjunct distinction.

As a pre-processing step, we used the logarithms of the frequencies instead of the frequencies themselves, to smooth the very skewed distributions that are typical to natural language. This has a down-scaling effect

[1] http://mi.cs.titech.ac.jp/kurihara/vdpmog.html

on extremely frequent features, without reducing them to the same scale as infrequent ones. Subsequently, the feature vector of each verb was normalized to unit length so that the frequency of the verb does not affect its representation.

Furthermore, dimensionality reduction was applied due to the large number of sparse features. The latter have similar distributions across verbs simply due to their sparsity. Since DPMMs do not weigh the features, a large number of sparse features is likely to influence inappropriately the clustering discovered. Nevertheless, sparse features incorporate useful semantic distinctions and have also performed well in some previous works. Therefore, rather than excluding them, we used principal component analysis (PCA) to reduce the dimensionality, employing the same cut-off point in all our experiments.

We evaluated the performance of the DPMMs in lexical-semantic clustering using the dataset of Sun et al. and experimented with various versions of the VALEX lexicon and the feature sets. In order to alleviate the effect of the random initialization, we ran each experiment 200 times. We achieved the best results with the cleanest version of the lexicon. Our performance was 69.5% homogeneity, 53.7% completeness and 60.5% V-measure, discovering 61.1 clusters on average. The best performance achieved in previous work was 59% in V-measure (Sun et al., 2008) using pairwise clustering (Puzicha et al., 2000). However, this result was achieved by setting the number of clusters to be discovered equal to the number of classes in the dataset, while DPMMs discover the number of clusters in the dataset.

5. Adding supervision

While the ability to discover novel information is attractive in NLP, in many cases it is also desirable to influence the solution with respect to some prior intuition or some considerations relevant to the application in mind. For example, while discovering finer-grained lexical-semantic classes than those included in the gold standard is useful, some NLP applications may benefit from a coarser clustering or a clustering targeted towards revealing some specific aspect of the dataset. For example, in the task of verb clustering, "encompass" and "carry" could be in the same cluster if the aim is to cluster all verbs meaning INCLUSION together, but they could also be separated if the aspect of MOTION of the latter is taken into account.

As an extension to this work, we implemented a semi-supervised version of the DPMMs that enables human supervision to guide the clustering solution. The human supervision is modelled as pairwise constraints over instances, as in Klein et al. (2002): given a pair of instances, either they should be clustered together (*must-link*) or not (*cannot-link*). This information can be obtained either from a human expert, or by appropriate manipulation of extant resources, such as ontologies. Specifying the relations between the instances results in an indirect labeling of the instances. Such labeling is likely to be re-usable, since it defines relations between the datapoints rather than explicit labels. The former are more likely to be useful to multiple tasks which might not have the same labels but could still take advantage of relations between datapoints.

The constraints will be added to the model by taking them into account during parameter estimation. We built a Dirichlet process mixture model using a standard sampling inference scheme (algorithm 3 from Neal (2000)). We chose the multinomial distribution to model the components. Following Neal (2000), we integrated analytically over the parameters θ_{z_i} of the model (Eq. 1 in Section 2).

In order to add supervision to the Dirichlet Process model we sample from distributions that respect the constraints imposed. In more detail, if two instances are connected with a *cannot-link* constraint, we will sample only from distributions that keep them in different components. Therefore, we set to 0 the probability of assigning an instance to a component containing *cannot-link* instance(s). Accordingly, in case they are connected with a *must-link* constraint, we sample only from distributions that keep them in the same component. Therefore, we set to 1 the probability of assigning an instance to a component containing *must-link* instance(s).

The expectation is that such constraints will not only affect the participating instances but the overall clustering as well. By guiding the clustering solution in this manner the DPMMs may discover knowledge better suited to the user's needs.

6. Experiments with supervision

In order to experiment with this method of adding supervision to the DPMMs, we implemented the DPMM model described in the previous section. The α parameter was determined by using a Gamma prior in Metropolis sampling scheme, which was run after each sampling of a component assignment z_i (Eq. 1).

In this second set of experiments we used the dataset of Korhonen et al. (2006b). It consists of 193 medium

to high frequency verbs from a corpus of 2230 full-text articles from 3 biomedical journals. The features, as in the Sun et al. (2008) dataset, were the subcategorization frames (SCFs) and their associated frequencies in the corpus, which were extracted automatically, resulting in 439 preposition-specific SCFs.

A team of domain experts and linguists were involved in creating a gold standard for this dataset. The former analyzed the verbs requiring domain-knowledge and the latter the general English and/or scientific ones. This effort resulted in a three-level gold standard which exemplifies the need for human supervision in order to influence the clustering solution discovered by the DPMMs, since ideally we would like to be able to discover any of these solutions. The number of classes was 16, 34 and 50 at each level of granularity.

As in Section 4, the feature set was very sparse and therefore we applied dimensionality reduction. However, PCA could not be used, since we used the multinomial distribution to model the components which cannot accept negative values. Therefore, we applied non-negative matrix factorization (NMF) (Lin, 2007) which decomposes the dataset in two dense matrices with non-negative values. In order to simulate the process of obtaining human supervision, we generated random verb pairs which we labelled as *must-link* or *cannot-link* according the version of the gold standard we aimed for.

We used 35 dimensions for the NMF dimensionality reduction. The base measure G_0 used was the normalized mean of the dataset, the initial value for the α was 1 and all the instances were assigned to a single component initially. We generated 100 pairs of verbs and obtained their *must-links* or *cannot-links* for each of the three level of granularity of the gold standard. First, we ran the DPMM without any supervision, in order to adapt itself to the data without any constraints for 100 iterations of the Gibbs sampler. Then, we ran the model using the constraints to restrict the sampling for another 100 iterations and obtained the final component assignment.

The results from these experiments appear in Table 1. The rows labeled "vanilla" contain the results for the standard unsupervised model. The other rows are labelled according to the version of the gold standard followed by the number of links obtained from it. The number of clusters discovered by all the versions of the model did not vary substantially, being between 37 and 41. It can be observed that adding supervision to the model guides it to clustering closer to the version of the gold standard the supervision was obtained from. For example, adding 100 links from the coarsest version which contains 16 classes (row "link16_100" in the "16 classes" part of the table) improves the V-measure by 2.28% when evaluating on the same version. However, when evaluating on the finest grained version of the standard (containing 50 classes), then the V-measure remains identical and only the homogeneity and completeness scores change. On the other hand, adding 100 links from the latter version of the gold standard, improves the performance by 1.39% when evaluating on it. As expected though, the performance drops for the 16-class version, since the supervision guides the clustering to a different solution. Adding supervision from the 34-class version, improves the performance by 1.76% in v-measure (row "link34_100"). Overall, the model is adapted towards the clustering solution aimed for. In the rows labeled "gauss" we report the result with the DPMM using Gaussians used in the experiments of Section 4, which discovered 63.23 clusters on average. The new model outperforms it at all level of gold standard, even without using supervision.

	hom	comp	V
16 classes			
vanilla	77.09%	64.11%	70%
link16_100	82.16%	64.52%	72.28%
link50_100	77.53%	62.69%	69.32%
gauss	78.54%	50.22%	61.26%
34 classes			
vanilla	70.24%	78.94%	74.34%
link34_100	73.19%	79.24&	76.10%
gauss	77.30%	66.79%	71.65%
50 classes			
vanilla	69.07%	87.43%	77.17%
link16_100	70.87%	84.71%	77.17%
link50_100	71.19%	87.63%	78.56%
gauss	76.53%	74.49%	75.49%

Table 1. Results on the biomedical verb dataset.

It must be noted that the different levels of granularity could have been achieved by appropriate tuning of the concentration parameter α (Eq. 1). However, to a non-expert in non-parametric modelling we believe it could be easier to simply provide examples of verbs that he or she would consider appropriate to be clustered together or separately. Moreover, α would affect the granularity of the clustering globally, while in a given application one might prefer to influence it more locally, something that can be achieved with the inclusion of pairwise links.

7. Conclusions - Future work

This paper makes several contributions. We applied DPMMs to a typical NLP learning task (lexical-semantic verb clustering) where the ability to discover the number of classes from the data is highly attractive. We experimented with two different datasets including (i) general English and (ii) biomedical verbs. Our quantitative evaluation using the recently introduced V-measure shows that the method compares favorably to earlier verb clustering methods which all rely on a pre-defined number of target clusters. In addition, we demonstrated how such models can be adapted to different needs using supervision in the form of pairwise links between instances.

The results encourage to apply DPMMs to further datasets and tasks. For verb clustering, we plan to investigate hierarchical Bayesian non-parametric models (Heller & Ghahramani, 2005) and to extend our experiments to larger datasets. We plan to conduct a thorough investigation of the ability of DPMMs to discover novel information not included in gold standards. Our preliminary assessment showed that many "errors" are due to the DPMM identifying verbs which are in fact too polysemous to be classified in single classes in large un-disambiguated input data and discovering semantically related classes as well as sub-classes of existing fine-grained classes. With respect to adding supervision to the model, we intend to explore ways in which the DPMM would select the links between instances to be labelled as in Klein et al. (2002), instead of obtaining them at random. Finally, an extrinsic evaluation of the clustering provided by DPMMs as part of an NLP application is likely to be very informative on their practical potential.

References

Briscoe, T., & Carroll, J. (2002). Robust accurate statistical annotation of general text. *Proceedings of the 3rd International Conference on Language Resources and Evaluation*.

Heller, K. A., & Ghahramani, Z. (2005). Bayesian hierarchical clustering. *Proceedings of the 22nd International Conference on Machine Learning*.

Klein, D., Kamvar, S., & Manning, C. (2002). From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. *Proceedings of the 19th International Conference on Machine Learning*.

Korhonen, A., Krymolowski, Y., & Briscoe, T. (2006a). A large subcategorization lexicon for natural language processing applications. *Proceedings of the 5th International Conference on Language Resources and Evaluation*.

Korhonen, A., Krymolowski, Y., & Collier, N. (2006b). Automatic classification of verbs in biomedical texts. *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*.

Kurihara, K., Welling, M., & Teh, Y. W. (2007). Collapsed variational Dirichlet process mixture models. *Proceedings of the 20th International Joint Conference on Artificial Intelligence*.

Levin, B. (1993). *English Verb Classes and Alternations: a preliminary investigation*. Chicago: University of Chicago Press.

Lin, C.-J. (2007). Projected gradient methods for non-negative matrix factorization. *Neural Computation*, *19*, 2756–2779.

Meilă, M. (2007). Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, *98*, 873–895.

Neal, R. M. (2000). Markov Chain Sampling Methods for Dirichlet Process Mixture Models. *Journal of Computational and Graphical Statistics*, *9*, 249–265.

Puzicha, J., Hofmann, T., & Buhmann, J. (2000). A theory of proximity based clustering: Structure detection by optimization. *Pattern Recognition*, *33*, 617–634.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, *66*, 846–850.

Rosenberg, A., & Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

Schulte im Walde, S. (2006). Experiments on the automatic induction of german semantic verb classes. *Computational Linguistics*, *32*, 159–194.

Sethuraman, J. (1994). A constructive definition of dirichlet priors. *Statistica Sinica*, *4*, 639–650.

Sun, L., Korhonen, A., & Krymolowski, Y. (2008). Verb class discovery from rich syntactic data. *Proceedings of the 9th International Conference on Intelligent Text Processing and Computational Linguistics*.