Coarse-to-Fine Inference and Learning for First-Order Probabilistic Models

Chloé Kiddon and **Pedro Domingos**
Department of Computer Science & Engineering
University of Washington
Seattle, WA 98105
{chloe,pedrod}@cs.washington.edu

Abstract

Coarse-to-fine approaches use sequences of increasingly fine approximations to control the complexity of inference and learning. These techniques are often used in NLP and vision applications. However, no coarse-to-fine inference or learning methods have been developed for general first-order probabilistic domains, where the potential gains are even higher. We present our Coarse-to-Fine Probabilistic Inference (CFPI) framework for general coarse-to-fine inference for first-order probabilistic models, which leverages a given or induced type hierarchy over objects in the domain. Starting by considering the inference problem at the coarsest type level, our approach performs inference at successively finer grains, pruning high- and low-probability atoms before refining. CFPI can be applied with any probabilistic inference method and can be used in both propositional and relational domains. CFPI provides theoretical guarantees on the errors incurred, and these guarantees can be tightened when CFPI is applied to specific inference algorithms. We also show how to learn parameters in a coarse-to-fine manner to maximize the efficiency of CFPI. We evaluate CFPI with the lifted belief propagation algorithm on social network link prediction and biomolecular event prediction tasks. These experiments show CFPI can greatly speed up inference without sacrificing accuracy.

Introduction

Probabilistic inference in AI problems is often intractable. Most widely used probabilistic representations in these problems are propositional, but in the last decade, many first-order probabilistic languages have been proposed (Getoor and Taskar 2007). Inference in these languages can be carried out by first converting to propositional form; however, more recently more efficient algorithms for lifted inference have been developed (Poole 2003; de Salvo Braz, Amir, and Roth 2007; Singla and Domingos 2008; Kersting, Ahmadi, and Natarajan 2009; Kisynski and Poole 2009). While lifting can yield large speedups over propositionalized inference, the blowup in the combinations of objects and relations still greatly limits its applicability. One solution is to perform approximate lifting, by grouping objects that behave similarly, even if they are not exactly alike (Singla 2009; Sen, Deshpande, and Getoor 2009; de Salvo Braz et al. 2009).

In this paper, we propose an approach to approximate lifting that scales much better that previous approaches by exploiting coarse-to-fine domain structure. Coarse-to-fine approaches are becoming more prevalent as probabilistic inference problems grow into larger, richer domains. The coarse-to-fine paradigm makes efficient inference possible while minimizing loss of accuracy. A coarse-to-fine approach performs inference at successively finer granularities. It uses the results from the coarser stages, where inference is faster, to guide and speed up inference at the more refined levels. A wide range of methods under the coarse-to-fine paradigm have been used in vision (e.g., Raphael (2001), Felzenszwalb and Huttenlocher (2006), Felzenszwalb, Girshick, and McAllester (2010), Weiss and Taskar (2010)), NLP (e.g., Petrov and Klein (2007), Carreras, Collins, and Koo (2008), Weiss and Taskar (2010)), and other fields. However, despite the growing interest in coarse-to-fine methods (e.g., Petrov et al. (2010)), no coarse-to-fine methods for general first-order probabilistic models have been proposed to date. Inference in these models could benefit greatly from the coarse-to-fine paradigm; the domains of these models tend to contain ontological structure where this type of approximation is applicable. The use of ontological information has been studied extensively, but almost entirely in the context of purely logical inference (Staab and Studer 2004). However, the need for it is arguably even greater in probabilistic inference.

Our Coarse-to-Fine Probabilistic Inference (CFPI) approach generalizes previous coarse-to-fine approaches in NLP etc., but also opens up many new applications. Given a type hierarchy, CFPI first performs inference using the coarsest type information, prunes atoms that are close to certain, then performs inference at the next finer level and repeats until the finest level is reached or the full query has been decided. (Alternatively, the type hierarchy itself could be induced from data.) CFPI is most efficient for models where pruning decisions can be made as early as possible. We describe our coarse-to-fine learning method that learns models optimized for CFPI by utilizing the type hierarchy; the lower levels refine the parameters at the higher levels, maximizing the gains.

CFPI treats coarse-to-fine inference as a succession of finer and finer applications of approximate lifted inference guided by a type hierarchy. CFPI can be applied with

Weighted First-Order Logic Rules	Evidence
1.4 $\texttt{TAs}(\texttt{x},\texttt{c}) \wedge \texttt{Teaches}(\texttt{y},\texttt{c}) \Rightarrow \texttt{Advises}(\texttt{y},\texttt{x})$	$\texttt{TAs}(\texttt{Anna},\texttt{AI101})$
4.3 $\texttt{Publication}(\texttt{t},\texttt{x}) \wedge \texttt{Advises}(\texttt{y},\texttt{x}) \Rightarrow \texttt{Publication}(\texttt{t},\texttt{y})$	

Table 1: Example of a Markov logic network and evidence. Free variables are implicitly universally quantified.

any probabilistic inference algorithm. Our framework uses and generalizes hierarchical models, which are widespread in machine learning and statistics (e.g., Gelman and Hill (2006), Pfeffer et al. (1999)). Our approach also incorporates many of the advantages of lazy inference (Poon, Domingos, and Sumner 2008).

Our algorithms are formulated in terms of Markov logic (Domingos and Lowd 2009). The generality and simplicity of Markov logic make it an attractive foundation for a coarse-to-fine inference and learning framework. In particular, our approach directly applies to all representations that are special cases of Markov logic, including standard graphical models, probabilistic context-free grammars, relational models, etc. However, our framework could also be formulated using other relational probabilistic languages.

We begin with necessary background, present the framework, and then provide bounds on the approximation error. We then report our experiments on two real-world domains (a social network one and a molecular biology one) applying CFPI with lifted belief propagation. Our results show that our approach can be more effective compared to lifted belief propagation without CFPI.

Background

Graphical models compactly represent the joint distribution of a set of variables $\mathbf{X} = (X_1, X_2, \ldots, X_n) \in \mathcal{X}$ as a product of factors (Pearl 1988): $P(\mathbf{X}{=}\mathbf{x}) = \frac{1}{Z}\prod_k f_k(\mathbf{x}_k)$, where each factor f_k is a non-negative function of a subset of the variables $\mathbf{x}_k$, and Z is a normalization constant. If $P(\mathbf{X}{=}\mathbf{x}) > 0$ for all x, the distribution can be equivalently represented as a *log-linear model*: $P(\mathbf{X}{=}\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_i w_i g_i(\mathbf{x})\right)$, where the *features* $g_i(\mathbf{x})$ are arbitrary functions of (a subset of) the state. The *factor graph* representation of a graphical model is a bipartite graph with a node for each variable and factor in the model (Kschischang, Frey, and Loeliger 2001). (For convenience, we consider one factor $f_i(\mathbf{x}) = \exp(w_i g_i(\mathbf{x}))$ per feature $g_i(\mathbf{x})$, i.e., we do not aggregate features over the same variables into a single factor.) Undirected edges connect variables with the appropriate factors. The main inference task in graphical models is to compute the conditional probability of some variables (the query) given the values of others (the evidence), by summing out the remaining variables. Inference methods for graphical models include belief propagation and MCMC.

A *first-order knowledge base (KB)* is a set of sentences or formulas in first-order logic. *Constants* represent objects in the domain of interest (e.g., people: `Amy`, `Bob`, etc.). *Variables* range over the set of constants. A *predicate* is a symbol that represents a relation among objects (e.g., `Advises`) or an attribute of an object (e.g., `Student`) and its arity (number of arguments it takes). An *atom* is a predicate applied to a tuple of variables or objects (e.g., $\texttt{Advises}(\texttt{Amy},\texttt{y})$) of the proper arity. A *clause* is a disjunction of atoms, each of which can either be negated or not. A *ground atom* is an atom with only constants as arguments. A *ground clause* is a disjunction of ground atoms or their negations.

First-order probabilistic languages combine graphical models with elements of first-order logic by defining template features that apply to whole classes of objects at once. A simple and powerful such language is *Markov logic* (Domingos and Lowd 2009). A *Markov logic network (MLN)* is a set of weighted first-order clauses. Given a set of constants, an MLN defines a Markov network with one node per ground atom and one feature per ground clause. The weight of a feature is the weight of the first-order clause that originated it. The probability of a state $\mathbf{x}$ is given by $P(\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_i w_i g_i(\mathbf{x})\right)$, where w_i is the weight of the i^{th} clause, $g_i = 1$ if the i^{th} clause is true, and $g_i = 0$ otherwise. Table 1 shows an example of a simple MLN representing an academia model. An example of a ground atom, given as evidence, is shown. States of the world where more advisees TA for their advisors, and advisees and their advisors coauthor publications, are more probable. Inference in Markov logic can be carried out by creating and running inference over the ground network, but this can be extremely inefficient because the size of the ground network is $O(d^c)$, where d is the number of objects in the domain and c is the highest clause arity. *Lifted* inference establishes a more compact version of the ground network in order to make inference more efficient. In *lifted belief propagation* (LBP), subsets of components in the ground network are identified that will send and receive identical messages during belief propagation (Singla and Domingos 2008).

Representation

The standard definition of an MLN assumes an undifferentiated set of constants. We begin by extending it to allow for a hierarchy of constant types.

Definition 1 A *type* is a set of constants $t = \{k_1, \ldots, k_n\}$. A type t is a *subtype* of another type t' iff $t \subset t'$. A type t is a *supertype* of another type t' iff $t' \subset t$. A *refinement* of a type t is a set of types $\{t_1, \ldots, t_m\}$ such that $\forall_{i,j}\ t_i \cap t_j = \emptyset$ and $t = t_1 \cup t_2 \cup \cdots \cup t_m$.

Definition 2 A *typed predicate* is a tuple $a = (a_0, t_1, \ldots, t_n)$, where a_0 is a predicate, n is a_0's arity, and t_i is the type of a_0's ith argument. A *typed atom* is a typed predicate applied to a tuple of variables or objects of the proper arity and types. A *typed clause* is a tuple $c = (c_0, t_1, \ldots, t_n)$, where c_0 is a first-order clause, n is the number of unique variables in c_0, and t_i is the type of the ith variable in c_0. The

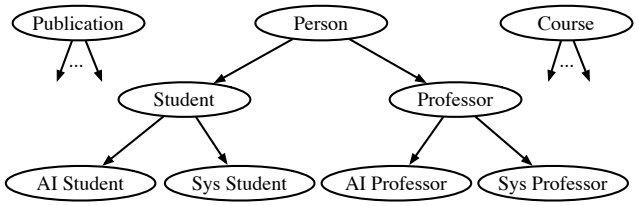

Figure 1: Example type hierarchy for an academia domain.

set of types in a typed predicate, atom, or clause is referred to as the predicate's, atom's, or clause's *type signature*.

Definition 3 A *typed MLN* $\mathbf{M}$ is a set of weighted typed clauses, $\{(c_i, w_i)\}$. It defines a ground Markov network with one node for each possible grounding of each typed atom in $\mathbf{M}$, and one feature for each possible grounding of each typed clause in $\mathbf{M}$ with constants from the corresponding types. The weight of a feature is the weight of the typed clause that originated it.

Definition 4 Given a set of types $\mathbf{T}$, $t_i \in \mathbf{T}$ is a *direct subtype* of $t_j \in \mathbf{T}$ iff $t_i \subset t_j$ and $\nexists t \in \mathbf{T}$ such that $t_i \subset t \subset t_j$. $\{t_1, t_2, \ldots, t_m\} \subset \mathbf{T}$ is a *direct refinement* of $t \in \mathbf{T}$ iff it is a refinement of t and $t_1, \ldots, t_m$ are direct subtypes of t. A set of types $\mathbf{T}$ is a *type hierarchy* iff within $\mathbf{T}$, each type has no subtypes or exactly one direct refinement, and $\forall_{i,j}\ (t_i \cap t_j = \emptyset) \vee (t_i \subset t_j) \vee (t_j \subset t_i)$. A *root type* has no supertypes; a *leaf type* has no subtypes.

A type hierarchy is a forest of types. It may be a tree, but an all-encompassing root type will usually be too general to be useful for inference. Figure 1 depicts an example type hierarchy for an academia domain.

Coarse-to-Fine Inference

Algorithm 1 shows pseudocode for the CFPI algorithm. It takes as input a type hierarchy $\mathbf{T}$, a typed MLN $\mathbf{M_T}$ over types in $\mathbf{T}$, a database of evidence $\mathbf{E}$, and a pruning threshold γ. CFPI begins by choosing an MLN $\mathbf{M}$ containing the weighted clauses in $\mathbf{M_T}$ whose type signatures are composed exclusively of the coarsest level types. These could be root types or a set of types from any cut of the type hierarchy. For example, in an academia domain, it may make more sense to consider students and professors separately from the start. CFPI then calls a pre-specified lifted probabilistic inference algorithm to compute the marginals of all the non-evidence ground atoms based on $\mathbf{M}$, the constants in $\mathbf{T}$, and the evidence $\mathbf{E}$. Ground atoms whose marginals are at most γ are added to the evidence as false, and those whose marginals are at least $1 - \gamma$ are added as true. The marginal probabilities of these pruned ground atoms are stored and returned in the output of CFPI. Any clauses now valid or unsatisfiable given the expanded evidence will not affect the results of subsequent inferences and are removed from $\mathbf{M}$.

CFPI then refines $\mathbf{M}$, replacing every clause c in $\mathbf{M}$ with the set of clauses obtained by direct refinement of the types in c's type signature. If v is a variable in c, v's type in a refined clause is a direct subtype of its type in c, and there is a refined clause for each possible combination of direct subtypes for the variables in c. Any leaf types are left unrefined. In general, it might be useful to refine some types and leave others unrefined, but this substantially increases the complexity of the algorithm and is left for future work. The clauses returned are the *direct clause refinements* of the clause c. The process ends when no more direct clause refinements are possible on the clauses in $\mathbf{M}$ or all ground atoms have been pruned; in either case, *Refine*$(\mathbf{M}, \mathbf{M_T})$ returns $\mathbf{M}$.

Algorithm 1 Coarse-to-Fine Probabilistic Inference

inputs: $\mathbf{M_T}$, a typed Markov logic network
$\mathbf{T}$, a type hierarchy
$\mathbf{E}$, a set of ground literals
γ, pruning threshold
calls: *Infer()*, a probabilistic inference algorithm
Refine(), a type refinement algorithm
$\mathbf{M} \leftarrow$ *Coarsest*$(\mathbf{M_T})$
repeat
  $P(\mathbf{x}|\mathbf{E}) \leftarrow$ *Infer*$(\mathbf{M}, \mathbf{T}, \mathbf{E})$
  for each atom x_i
    if $P(x_i|\mathbf{E}) \leq \gamma$ **then** $\mathbf{E} \leftarrow \mathbf{E} \cup \{\neg x_i\}$
    else if $P(x_i|\mathbf{E}) \geq 1 - \gamma$ **then** $\mathbf{E} \leftarrow \mathbf{E} \cup \{x_i\}$
  $\mathbf{M} \leftarrow \mathbf{M} \setminus \{$valid and unsatisfiable clauses under $\mathbf{E}\}$
  $\mathbf{M} \leftarrow$ *Refine*$(\mathbf{M}, \mathbf{M_T})$
until *Refine*$(\mathbf{M}, \mathbf{M_T}) = \mathbf{M}$
$P(\mathbf{x}|\mathbf{E}) \leftarrow$ *Infer*$(\mathbf{M}, \mathbf{T}, \mathbf{E})$

Previous coarse-to-fine approaches can be cast into this general framework. For example, in Petrov and Klein (2007), the type hierarchy is the hierarchy of nonterminals, the refinement procedure is the reverse projection of the set of coarse-to-fine grammars, and inference is the inside-outside algorithm.

At every step, the MLN grows by refining clauses, but also shrinks by pruning. The goal is to contain the complexity of inference, while keeping it focused on where it is most needed: the ground atoms we are most uncertain about. The following theorem bounds the approximation error incurred by this process, relative to exact inference.

Theorem 1 *Let γ be the CFPI pruning threshold, n_k be the number of atoms pruned at level k, m be the total number of features, $\Delta_k w$ be the maximum error in weights at level k, δ be the maximum absolute error in the marginal probabilities returned by Infer(), $P(x_i|E)$ be the true marginal probability of x_i given evidence E, $\hat{P}_k(x_i|E)$ be the approximate marginal probability of x_i returned by CFPI at level k given evidence E, and K be the level at which CFPI stops. If x_i is pruned at level k, the error in the probability of x_i, $\epsilon(x_i) = |\hat{P}_k(x_i|E) - P(x_i|E)|$, is bounded by*

$$\epsilon(x_i) \leq (\gamma + \delta)\left(e^{2m\Delta_k w} + \sum_{i=1}^{k-1} n_i e^{2m\Delta_i w}\right).$$

If x_i is not pruned,

$$\epsilon(x_i) \leq (\gamma + \delta)\left(\sum_{i=1}^{K-1} n_i e^{2m\Delta_i w}(1 + \hat{P}_K(x_i|E))\right).$$

(Proofs of theorems are provided in the appendix.) When atoms are pruned, the set of possible worlds shrinks and the probabilities of the remaining possible worlds must be renormalized. Intuitively, errors stem from pruning possible worlds that have non-zero probability (or pruning worlds where $P(\mathbf{x}) \neq 1$ for the high-probability case). We can bound the probability mass of pruned worlds based on weight approximations and the number of previously pruned atoms. In turn, we can use those bounds to bound errors in atom marginals.

Infer() can be any lifted probabilistic inference algorithm (or even propositionalization followed by ground inference, although this is unlikely to scale even in the context of CFPI). If the inference algorithm is exact (e.g., FOVE (de Salvo Braz, Amir, and Roth 2007)), the error $\delta = 0$ in the above bound. However, realistic domains generally require approximate inference.

In this paper, we use lifted belief propagation (Singla and Domingos 2008). We call CFPI applied to lifted belief propagation CFLBP (Coarse-to-Fine Lifted Belief Propagation). We now provide an error bound for CFLBP. While Theorem 1 provides an intuitive error bound that is independent of the inference method used with CFPI, Theorem 2 provides a tighter bound when the error is calculated concurrently with inference. We base our algorithm on Theorem 15 of Ihler et al. (2005) that bounds errors on atom marginals due to multiplicative errors in the messages passed during BP. Since lifted BP computes the same marginals as ground BP, for the purposes of a proof, the former can be treated as the latter. We can view the errors in the messages passed during BP in level k of CFLBP as multiplicative errors on the messages from factors to nodes at each step of BP, due to weight approximations at that level and the loss of pruned atoms.

Theorem 2 *For the network at level k of CFPI, let p_x^k be the probability estimated by BP at convergence, $\hat{p}_x^k$ be the probability estimated by CFLBP after n iterations of BP, σ^- and σ^+ be the sets of low- and high-probability atoms pruned in CFLBP's previous $k-1$ runs of BP, α_f^k be the difference in weight of factor f between level k and the final level K, and γ be the pruning threshold. For a binary node x, p_x^k can be bounded as follows:*
For $x \in \sigma^-$: $\quad 0 \leq p_x^k \leq \gamma$
For $x \in \sigma^+$: $\quad 1-\gamma \leq p_x^k \leq 1$
And for $x \notin \sigma^- \cup \sigma^+$:

$$p_x^k \geq \frac{1}{(\xi_x^{k,n})^2[(1/\hat{p}_x^k)-1]+1} = lb(p_x^k) \quad \textit{and}$$
$$p_x^k \leq \frac{1}{(1/\xi_x^{k,n})^2[(1/\hat{p}_x^k)-1]+1} = ub(p_x^k),$$

where

$$\begin{aligned}
\log \xi_x^{k,n} &= \sum_{f \in nb(x)} \log \nu_{f,x}^{k,n}, \\
\nu_{x,f}^{k,1} &= d(f)^2, \\
\log \nu_{x,f}^{k,i+1} &= \sum_{h \in nb(x) \setminus \{f\}} \log \nu_{h,x}^{k,i}, \\
\log \nu_{f,x}^{k,i+1} &= \log \frac{d(f)^2 \tau_{f,x}^{k,i}+1}{d(f)^2+\tau_{f,x}^{k,i}} + \log d(\varepsilon_{f,x}^k), \\
\log \tau_{f,x}^{k,i} &= \sum_{y \in nb(f) \setminus \{x\}} \log \nu_{y,f}^{k,i}, \\
d(\varepsilon_{f,x}^k) &= \gamma^{-\frac{1}{2}|\sigma_f^-|}(1-\gamma)^{-\frac{1}{2}|\sigma_f^+|} e^{\frac{1}{2}\alpha_f^k},
\end{aligned}$$

and nodes are only pruned at level k' when either $ub(p_x^{k'}) \leq \gamma$ or $lb(p_x^{k'}) \geq 1-\gamma$.

Although Theorem 2 does not have a particularly intuitive form, it yields much tighter bounds than Theorem 1 if we perform the bound computations as we run BP. If no atoms are pruned at previous levels, the fixed point beliefs returned from CFLBP on its k^{th} level of BP after n iterations will be equivalent to those returned by BP after n iterations on the network at that level.

Coarse-to-Fine Learning

The critical assumption invoked by the inference framework is that objects of the same type tend to act in similar manners. In terms of a typed MLN, stronger weights on clauses over coarser types allow pruning decisions to be made earlier, which speeds up later iterations of inference. To achieve models of this type, we learn weights in a coarse-to-fine manner through a series of successive refinements of clauses. The weights for clauses at each iteration of learning are learned with all weights learned from preceding iterations held fixed. The effect is that a weight learned for a typed clause is the additional weight given to a clause grounding based on having that new type information. As the weights are learned for clauses over finer and finer type signatures, these weights should become successively smaller as the extra type information is less important. A benefit of this coarse-to-fine approach to learning is that as soon as refining a typed clause does not give any new information (e.g., all direct refinements of a clause are learned to have 0 weight), the typed clause need not be refined further. The result is a sparser model that will correspond to fewer possible refinements during the inference process and therefore more efficient inference.

Proposition 1 *For a typed MLN $\mathbf{M_T}$ learned in the coarse-to-fine framework, there is an equivalent* typed-flattened *MLN $\mathbf{M'_T}$ such that no clause $c \in \mathbf{M'_T}$ can be obtained through a series of direct clause refinements of any other clause $c' \in \mathbf{M'_T}$.*

When *Refine*$(\mathbf{M}, \mathbf{M_T})$ replaces a clause c in $\mathbf{M}$ by its direct clause refinements, the weight of each new typed clause c'_i added to $\mathbf{M}$ is $w + w'_i$, where w is the weight of c in $\mathbf{M}$ and w'_i is the weight of c'_i in $\mathbf{M_T}$. When there are no more refinements, the resulting typed MLN will be a subset of the type-flattened MLN $\mathbf{M'_T}$, accounting for pruned clauses.

Coarse-to-fine learning is not essential, but it greatly improves the efficiency of coarse-to-fine inference. By design it yields a model that is equivalent to the type-flattened one, and so incurs no loss in accuracy. We note that using regularization while learning causes the typed MLN to only be approximately equivalent to the type-flattened MLN but can

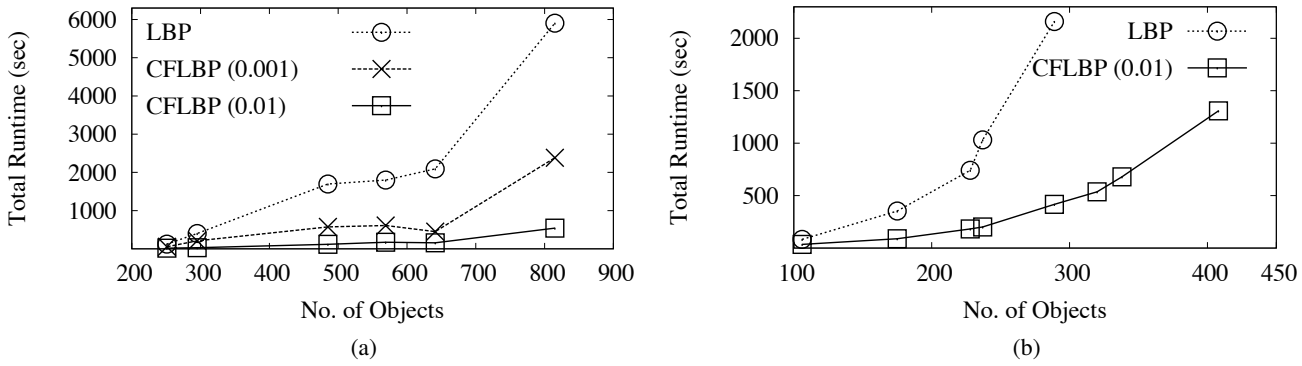

Figure 2: Total runtime of algorithms over (a) the UW CSE and (b) the GENIA data sets.

UW-CSE data set

Algorithm	Pruning Threshold	Init	Infer	Prune	Avg. CLL	# Superfeatures
LBP	N/A	3441.59	2457.34	N/A	-0.00433	8.2 million
CFLBP	0.001	2172.45	208.59	0.99	-0.00433	485,507
CFLBP	0.01	537.93	2.08	1.15	-0.00431	10,328

GENIA data set

Algorithm	Pruning Threshold	Init	Infer	Prune	Avg. CLL	# Superfeatures
LBP	N/A	1305.08	846.21	N/A	-0.01062	8.5 million
CFLBP	0.01	415.31	0.40	0.36	-0.01102	3,478

Table 2: Results for the full UW-CSE data set and GENIA data set over 150 abstracts. For CFLBP, number of superfeatures counts the most used during any level. *Init*, *Infer*, and *Prune* times given in seconds.

improve accuracy in the same way that hierarchical smoothing does.

Experiments

We experimented on a link prediction task in a social networking domain and an event prediction task in a molecular biology domain to compare the running time and accuracy of lifted belief propagation (LBP) applied with and without the CFPI framework. In both tasks, we assumed that all type information was known. That is, given a type hierarchy $\mathbf{T}$, each object is assigned a set of types $\{t_1, \ldots, t_n\} \subset \mathbf{T}$ where t_1 is a root type, t_i is a direct subtype of t_{i-1} for all $i > 1$, and t_n is a leaf type. CFPI can be applied in cases with incomplete type information, an experiment left for future work. We implemented CFLBP as an extension of the open-source Alchemy system (Kok et al. 2007). Currently, Alchemy does not allow for duplicate clauses with different type signatures. Instead we added type predicates to the formulas in our model to denote the correct type signatures. We compared running CFLBP over a typed MLN to running LBP over the equivalent type-flattened MLN. We ran each algorithm until either it converged or the number of iterations exceeded 100. We did not require each algorithm to run for the full 100 iterations since the network shrinkage that occurs with CFLBP may allow it to converge faster and is an integral part of its efficiency.

In our experiments, we used regularization when learning the models. We used L1-regularization when learning the model for the the link prediction task and L2-regularization when learning the event prediction task's model; future work will include a more thorough analysis of how using different regularization techniques during learning affects the speed and accuracy of CFPI.

Link Prediction

The ability to predict connections between objects is very important in a variety of domains such as social network analysis, bibliometrics, and micro-biology protein interactions. We experimented on the link prediction task of Richardson and Domingos (2009), using the UW-CSE database that is publicly available on the Alchemy project website[1]. The task is to predict the `AdvisedBy` relation given evidence on teaching assignments, publication records, etc. We manually created a type hierarchy that corresponded well to the domain. The *Person* type is split into a *Professor* and a *Student* type, both of which are split further by area (e.g., AI, Systems); the *Student* type is split further by point in the graduate program (e.g., Pre-Quals, Post-Generals). The *Class* type is split by area followed by level. We tested on 43 of the 94 formulas in the UW-CSE MLN; we removed formulas with existential quantifiers and duplicates that remained after the removal of "type" predicates such as `Student(x)`. The type-flattened MLN had 10,150 typed clauses from matching the 43 formulas with varying

[1] http://alchemy.cs.washington.edu

type signatures. The full database contains ∼4,000 predicate groundings, including type predicates. To evaluate inference over different numbers of objects in the domain, we randomly selected graph cuts of various sizes from the domain. Figure 2(a) shows a comparison of the runtimes of CFLBP and LBP for different sized cuts of the UW-CSE data set. We ran CFLBP with pruning thresholds of $\gamma = 0.01$ and $\gamma = 0.001$. The time is the sum of both initialization of the network and the inference itself; the times for CFLBP also include the refinement times after each level. For each cut of the UW-CSE data set, the average conditional log likelihood (CLL) of the results returned by CFLBP with either pruning threshold were virtually the same as the average conditional log likelihood returned by LBP. Table 2 summarizes the results of the UW-CSE link prediction experiment over the full UW CSE data set. The full data set contained 815 objects, including 265 people, and 3833 evidence predicates. With $\gamma = 0.01$, we achieve an order of magnitude speedup.

Biomolecular Event Prediction

As new biomedical literature accumulates at a rapid pace, the importance of text mining systems tailored to the domain of molecular biology is increasing. One important task is the identification and extraction of biomolecular events from text. Event prediction is a challenging task (Kim et al. 2003) and is not the focus of this paper. Our simplified task is to predict which entities are the causes and themes of identified events contained in the text, represented by two predicates: $\texttt{Cause}(\texttt{event}, \texttt{entity})$ and $\texttt{Theme}(\texttt{event}, \texttt{entity})$. We used the GENIA event corpus that marks linguistic expressions that identify biomedical events in scientific literature spanning 1,000 Medline abstracts; there are 36,114 events labeled, and the corpus contains a full type hierarchy of 32 entity types and 28 event types (Kim, Ohta, and Tsujii 2008). Our features include semantic co-occurrence and direct semantic dependencies with a set of key stems (e.g., $\texttt{Subj}(\texttt{entity}, \texttt{stem}, \texttt{event})$). We also learned global features that represent the roles that certain entities tend to fill. We used the Stanford parser[2] for dependency parsing and a Porter stemmer to identify key stems.[3] We restricted our focus to events with one cause and one theme or no cause and two themes where we could extract interesting semantic information at our simple level. The model was learned over half the GENIA event corpus and tested on the other half; abstract samples of varying sizes were randomly generated. From 13 untyped clauses, the type-flattened MLN had 38,020 clauses.

Figure 2(b) shows a comparison of the runtimes of CFLBP with $\gamma = 0.01$ and LBP. For each test set where both CFLBP and LBP finished, the average conditional log likelihoods were almost identical. The largest difference in average conditional log likelihood was 0.019 with a dataset of 175 objects; in all other tests, the difference between the averages was never more than 0.001. Table 2 summarizes the results of the the largest GENIA event prediction experiment where both LBP and CFLBP finished without running out of memory. This test set included 125 events and 164 entities.

[2]http://nlp.stanford.edu/software/lex-parser.shtml

[3]http://tartarus.org/∼martin/PorterStemmer

Conclusion and Future Work

We presented a general framework for coarse-to-fine inference and learning. We provided bounds on the approximation error incurred by this framework. We also proposed a simple weight learning method that maximizes the gains obtainable by this type of inference. Experiments on two domains show the benefits of our approach. Directions for future work include: inducing the type hierarchy from data for use in CFPI; broadening the types of type structure allowed by CFPI (e.g., multiple inheritance); and applying CFPI to other lifted probabilistic inference algorithms besides LBP.

Acknowledgements

We thank Aniruddh Nath for his help with Theorem 2. This research was partly funded by ARO grant W911NF-08-1-0242, AFRL contract FA8750-09-C-0181, NSF grant IIS-0803481, ONR grant N00014-08-1-0670, and the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0718124. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, DARPA, AFRL, NSF, ONR, or the United States Government.

Appendix: Proofs of Theorems

Proof of Theorem 1

The probability of an atom is the probability of all the worlds (e.g., atom assignments $\mathbf{x}$) in which that atom is true:

$$P(x_i) = \sum_{\mathbf{x} \in \mathcal{X}} \mathbf{x}_i P(\mathbf{x})$$

where $\mathbf{x}_i$ is 1 if x_i is true in $\mathbf{x}$ and 0 otherwise. Assume that x_i is pruned at level k if its approximate marginal probability, $\hat{P}_k(x_i)$, falls below γ after running inference at level k. (We will consider pruning high-probability atoms later.) If x_i is pruned, then the probability of all the worlds where x_i is true is set to 0; these worlds are essentially pruned from the set of possible worlds.

Let $\mathcal{W}$ be a set of worlds. Let $P'(x_i)$ be the marginal probability of x_i given that the worlds in $\mathcal{W}$ have been pruned (e.g., the probability of each world in $\mathcal{W}$ is set to 0). Then,

$$P'(x_i) = \frac{\sum_{\mathbf{x} \in \mathcal{X} \setminus \mathcal{W}} \mathbf{x}_i e^{\sum_j w_j f_j(\mathbf{x})}}{\sum_{\mathbf{x} \in \mathcal{X} \setminus \mathcal{W}} e^{\sum_j w_j f_j(\mathbf{x})}}.$$

At each level k, the weight w_j is approximated by some $\hat{w}_j$ with at most difference $\Delta_k w$:

$$|w_j - \hat{w}_j| \leq \Delta_k w.$$

Assume now that $\mathcal{W}$ is the set of worlds that have been pruned in levels 1 through $k-1$. If x_i is pruned at level

k,

$$\begin{aligned} P'(x_i) &\leq \frac{\sum_{\mathbf{x}\in\mathcal{X}\setminus\mathcal{W}} \mathbf{x}_i e^{\sum_j (\hat{w}_j+\Delta_k w) f_j(\mathbf{x})}}{\sum_{\mathbf{x}\in\mathcal{X}\setminus\mathcal{W}} e^{\sum_j (\hat{w}_j-\Delta_k w) f_j(\mathbf{x})}} \\ &\leq \frac{\sum_{\mathbf{x}\in\mathcal{X}\setminus\mathcal{W}} \mathbf{x}_i e^{\sum_j \hat{w}_j f_j(\mathbf{x})}}{\sum_{\mathbf{x}\in\mathcal{X}\setminus\mathcal{W}} e^{\sum_j \hat{w}_j f_j(\mathbf{x})}} \frac{e^{m\Delta_k w}}{e^{-m\Delta_k w}} \\ &\leq \hat{P}_k(x_i) e^{2m\Delta_k w} \\ &\leq \gamma e^{2m\Delta_k w} = \gamma_k, \end{aligned}$$

where m is the total number of features.

When atoms are pruned, worlds are pruned, and the probability of the unpruned worlds has to be renormalized to compensate. Let $\mathcal{W}_k$ be the set of worlds pruned at level k. If $\mathbf{x}$ is an unpruned world after inference at level k, the new probability of $\mathbf{x}$, $P'(\mathbf{x})$, is

$$P'(\mathbf{x}) = \frac{P(\mathbf{x})}{1 - \sum_{l=1}^{k} P(\mathcal{W}_l)}. \tag{1}$$

Since $0 \leq P(\mathcal{W}_l) < 1$, $P'(\mathbf{x}) \geq P(\mathbf{x})$.

By the union bound, the probability of $\mathcal{W}_k$ (e.g., the probability of the union of all worlds where at least one pruned atom is true) is bounded by the sum of the probabilities of each pruned atom:

$$P(\mathcal{W}_k) \leq P'(\mathcal{W}_k) \leq n_k \gamma_k$$

where n_k is the number of atoms pruned at level k. Therefore, using Equation 1 we bound $P(\mathbf{x})$ as such:

$$\left(1 - \sum_{l=1}^{k} n_l \gamma_l\right) P'(\mathbf{x}) \leq P(\mathbf{x}) \leq P'(\mathbf{x}).$$

Since this is true for any world $\mathbf{x}$, it is also true for any set of worlds. If x_i is pruned at level k, $0 \leq \hat{P}_k(x_i) \leq \gamma$ and

$$\begin{aligned} 0 \leq P(x_i) &\leq \sum_{\mathbf{x}\in\mathcal{X}\setminus\mathcal{W}} \mathbf{x}_i P'(\mathbf{x}) + \sum_{\mathbf{x}\in\mathcal{W}} \mathbf{x}_i P'(\mathbf{x}) \\ &\leq \gamma e^{2m\Delta_k w} + \sum_{i=1}^{k-1} n_i \gamma e^{2m\Delta_i w}. \end{aligned}$$

Computing the bound for an atom pruned for having high probability is the equivalent to computing the bound for the negation of that atom for having low probability. If $P'(\bar{x}_i) \leq \gamma_k$, $P'(x_i) \geq 1 - \gamma_k$. While the computation follows similarly, the error bound is slightly tighter than the one for low-probability atoms. For simplicity, we provided the tightest bound that covers both cases. After conditioning everything on evidence E, factoring out the γ, and adding in δ to take into account errors from the inference algorithm, the first equation in Theorem 1 follows.

If an atom x_i is not pruned during the course of CFPI, then without bounding $\hat{P}_K(x_i)$, we use the same type of computation as we used for pruned atoms to get the second equation in Theorem 1. □

Proof of Theorem 2

The *dynamic range* of a function is defined as follows (Ihler, Fisher, and Willsky 2005):

$$d(f) = \sup_{x,y} \sqrt{f(x)/f(y)}.$$

At each refinement level, the messages in BP have errors introduced from the approximation of the factor weights from the coarse type signature and from the loss of messages from pruned nodes at earlier levels of refinement. At a level k, the difference between the weight of a factor f, w_f^k, and the true weight of the factor, w_f^K, at the finest level is $\alpha_f^k = w_f^k - w_f^K$, the error in the outgoing message is bounded by $e^{\alpha_f^k}$. The error reaches this bound when all possible states are compatible with the factor; in practice, the error will be much smaller.

Assuming that in levels 1 through $k-1$ we did not prune any node whose true probability is outside the pruning threshold γ. Then, the bound on the error of an incoming message from a pruned low node is γ, and the bound on the error of a message from a pruned high node is $1-\gamma$. If σ_f^- is the set of nodes neighboring a factor f that have been pruned for having a low probability, and σ_f^+ is the set of nodes neighboring f that were pruned for having a high probability, the multiplicative error of the messages from a factor f to a unpruned node x from the weight approximation and the pruned nodes is:

$$\varepsilon_{f,x}^k = e^{\alpha_f^k} \gamma^{-|\sigma_f^-|} (1-\gamma)^{-|\sigma_f^+|}.$$

Therefore, the dynamic range of the error is:

$$\begin{aligned} d(\varepsilon_{f,x}^k) &= \sqrt{e^{\alpha_f^k} \gamma^{-|\sigma_f^-|} (1-\gamma)^{-|\sigma_f^+|} / 1} \\ &= \gamma^{-\frac{1}{2}|\sigma_f^-|} (1-\gamma)^{-\frac{1}{2}|\sigma_f^+|} e^{\frac{1}{2}\alpha_f^k}. \end{aligned}$$

Theorem 15 of Ihler et al. (2005) implies that, for any fixed point beliefs $\{M_x\}$ found by BP, after $n \geq 1$ iterations of BP at level k of CFLBP resulting in beliefs $\{\hat{M}_x^{k,n}\}$ we have:

$$\log d(M_x / \hat{M}_x^{k,n}) \leq \sum_{f\in nb(x)} \log \nu_{f,x}^{k,n} = \log \xi_x^{k,n}.$$

It follows that $d(M_x \hat{M}_x^{k,n}) \leq \xi_x^{k,n}$, and therefore $\frac{M_x(1)/\hat{M}_x^{k,n}(1)}{M_x(0)/\hat{M}_x^{k,n}(0)} \leq (\xi_x^{k,n})^2$ and $(1-\hat{p}_x)/\hat{p}_x \leq (\xi_x^{k,n})^2 (1-p_x)/p_x$, where p_x and $\hat{p}_x$ are obtained by normalizing M_x and $\hat{M}_x$. The upper bound follows, and the lower bound can be obtained similarly. □

References

Carreras, X.; Collins, M.; and Koo, T. 2008. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, 9–16.

de Salvo Braz, R.; Amir, E.; and Roth, D. 2007. Lifted first-order probabilistic inference. In Getoor, L., and Taskar, B., eds., *Introduction to Statistical Relational Learning*. MIT Press. 433–450.

de Salvo Braz, R.; Natarajan, S.; Bui, H.; Shavlik, J.; and Russell, S. 2009. Anytime lifted belief propagation. In *Proceedings of the Sixth International Workshop on Statistical Relational Learning*.

Domingos, P., and Lowd, D. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan Kaufmann.

Felzenszwalb, P. F., and Huttenlocher, D. P. 2006. Efficient belief propagation for early vision. *International Journal of Computer Vision* 70(1):41–54.

Felzenszwalb, P.; Girshick, R.; and McAllester, D. 2010. Cascade object detection with deformable part models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2241–2248.

Gelman, A., and Hill, J. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

Getoor, L., and Taskar, B., eds. 2007. *Introduction to Statistical Relational Learning*. MIT Press.

Ihler, A. T.; III, J. W. F.; and Willsky, A. S. 2005. Loopy belief propagation: Convergence and effects of message errors. *Journal of Machine Learning Research* 6:905–936.

Kersting, K.; Ahmadi, B.; and Natarajan, S. 2009. Counting belief propagation. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, 277–284.

Kim, J.-D.; Ohta, T.; Tateisi, Y.; and Tsujii, J. 2003. GENIA corpus–semantically annotated corpus for bio-textmining. *Bioinformatics* 19(1):i180–i182.

Kim, J.-D.; Ohta, T.; and Tsujii, J. 2008. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics* 9(1):10.

Kisynski, J., and Poole, D. 2009. Lifted aggregation in directed first-order probabilistic models. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, 1922–1929.

Kok, S.; Sumner, M.; Richardson, M.; Singla, P.; Lowd, H. P. D.; and Domingos, P. 2007. The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington, Seattle, WA. http://alchemy.cs.washington.edu.

Kschischang, F. R.; Frey, B. J.; and Loeliger, H.-A. 2001. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory* 47(2):498–519.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

Petrov, S., and Klein, D. 2007. Learning and inference for hierarchically split PCFGs. In *Proceedings of the Twenty-Second National Conference on Artificial Intelligence*, 1663–1666.

Petrov, S.; Sapp, B.; Taskar, B.; and Weiss, D. (organizers). 2010. NIPS 2010 Workshop on Coarse-to-Fine Learning and Inference. Whistler, B.C.

Pfeffer, A.; Koller, D.; Milch, B.; and Takusagawa, K. T. 1999. SPOOK: A system for probabilistic object-oriented knowledge representation. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 541–550.

Poole, D. 2003. First-order probabilistic inference. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, 985–991.

Poon, H.; Domingos, P.; and Sumner, M. 2008. A general method for reducing the complexity of relational inference and its application to MCMC. In *Proceedings of the Twenty-Third National Conference on Artificial Intelligence*, 1075–1080.

Raphael, C. 2001. Coarse-to-fine dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23(12):1379–1390.

Sen, P.; Deshpande, A.; and Getoor, L. 2009. Bisimulation-based approximate lifted inference. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 496–505.

Singla, P., and Domingos, P. 2008. Lifted first-order belief propagation. In *Proceedings of the Twenty-Third National Conference on Artificial Intelligence*, 1094–1099.

Singla, P. 2009. *Markov Logic: Theory, Algorithms and Applications*. PhD in Computer Science & Engineering, University of Washington, Seattle, WA.

Staab, S., and Studer, R. 2004. *Handbook on Ontologies (International Handbooks on Information Systems)*. SpringerVerlag.

Weiss, D., and Taskar, B. 2010. Structured prediction cascades. In *International Conference on Artificial Intelligence and Statistics*, 916–923.