Margin-based Decomposed Amortized Inference

Gourab Kundu* and **Vivek Srikumar*** and **Dan Roth**
University of Illinois, Urbana-Champaign
Urbana, IL. 61801
{kundu2, vsrikum2, danr}@illinois.edu

Abstract

Given that structured output prediction is typically performed over entire datasets, one natural question is whether it is possible to re-use computation from earlier inference instances to speed up inference for future instances. Amortized inference has been proposed as a way to accomplish this. In this paper, first, we introduce a new amortized inference algorithm called the *Margin-based Amortized Inference*, which uses the notion of structured margin to identify inference problems for which previous solutions are provably optimal. Second, we introduce *decomposed amortized inference*, which is designed to address very large inference problems, where earlier amortization methods become less effective. This approach works by decomposing the output structure and applying amortization piece-wise, thus increasing the chance that we can re-use previous solutions for *parts* of the output structure. These parts are then combined to a global coherent solution using Lagrangian relaxation. In our experiments, using the NLP tasks of semantic role labeling and entity-relation extraction, we demonstrate that with the margin-based algorithm, we need to call the inference engine only for a third of the test examples. Further, we show that the decomposed variant of margin-based amortized inference achieves a greater reduction in the number of inference calls.

1 Introduction

A wide variety of NLP problems can be naturally cast as structured prediction problems. For some structures like sequences or parse trees, specialized and tractable dynamic programming algorithms have proven to be very effective. However, as the structures under consideration become increasingly complex, the computational problem of predicting structures can become very expensive, and in the worst case, intractable.

In this paper, we focus on an inference technique called *amortized inference* (Srikumar et al., 2012), where previous solutions to inference problems are used to speed up new instances. The main observation that leads to amortized inference is that, very often, for different examples of the same size, the structures that maximize the score are identical. If we can efficiently identify that two inference problems have the same solution, then we can re-use previously computed structures for newer examples, thus giving us a speedup.

This paper has two contributions. First, we describe a novel algorithm for amortized inference called *margin-based amortization*. This algorithm is on an examination of the structured margin of a prediction. For a new inference problem, if this margin is larger than the sum of the decrease in the score of the previous prediction *and* any increase in the score of the second best one, then the previous solution will be the highest scoring one for the new problem. We formalize this intuition to derive an algorithm that finds provably optimal solutions and show that this approach is a generalization of previously identified schemes (based on Theorem 1 of (Srikumar et al., 2012)).

Second, we argue that the idea of amortization is best exploited at the level of *parts* of the structures rather than the entire structure because we expect a much higher redundancy in the parts. We introduce the notion of *decomposed amortized inference*, whereby we can attain a significant improvement in speedup by considering repeated sub-structures across the dataset and applying any amortized inference algorithm for the parts.

* These authors contributed equally to this work.

We evaluate the two schemes and their combination on two NLP tasks where the output is encoded as a structure: PropBank semantic role labeling (Punyakanok et al., 2008) and the problem of recognizing entities and relations in text (Roth and Yih, 2007; Kate and Mooney, 2010). In these problems, the inference problem has been framed as an integer linear program (ILP). We compare our methods with previous amortized inference methods and show that margin-based amortization combined with decomposition significantly outperforms existing methods.

2 Problem Definition and Notation

Structured output prediction encompasses a wide variety of NLP problems like part-of-speech tagging, parsing and machine translation. The language of 0-1 integer linear programs (ILP) provides a convenient analytical tool for representing structured prediction problems. The general setting consists of binary inference variables each of which is associated with a score. The goal of inference is to find the highest scoring global assignment of the variables from a feasible set of assignments, which is defined by linear inequalities.

While efficient inference algorithms exist for special families of structures (like linear chains and trees), in the general case, inference can be computationally intractable. One approach to deal with the computational complexity of inference is to use an off-the-shelf ILP solver for solving the inference problem. This approach has seen increasing use in the NLP community over the last several years (for example, (Roth and Yih, 2004; Clarke and Lapata, 2006; Riedel and Clarke, 2006) and many others). Other approaches for solving inference include the use of cutting plane inference (Riedel, 2009), dual decomposition (Koo et al., 2010; Rush et al., 2010) and the related method of Lagrangian relaxation (Rush and Collins, 2011; Chang and Collins, 2011).

(Srikumar et al., 2012) introduced the notion of an *amortized inference algorithm*, defined as an inference algorithm that can use previous predictions to speed up inference time, thereby giving an amortized gain in inference time over the lifetime of the program.

The motivation for amortized inference comes from the observation that though the number of possible structures could be large, in practice, only a small number of these are ever seen in real data. Furthermore, among the observed structures, a small subset typically occurs much more frequently than the others. Figure 1 illustrates this observation in the context of part-of-speech tagging. If we can efficiently characterize and identify inference instances that have the same solution, we can take advantage of previously performed computation without paying the high computational cost of inference.

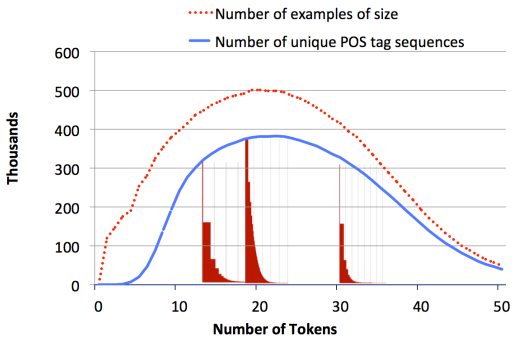

Figure 1: Comparison of number of instances and the number of unique observed part-of-speech structures in the Gigaword corpus. Note that the number of observed structures (blue solid line) is much lower than the number of sentences (red dotted line) for all sentence lengths, with the difference being very pronounced for shorter sentences. Embedded in the graph are three histograms that show the distribution of observed structures for sentences of length 15, 20 and 30. In all cases, we see that a small number of tag sequences are much more frequent than the others.

We denote inference problems by the bold-faced letters $\mathbf{p}$ and $\mathbf{q}$. For a problem $\mathbf{p}$, the goal of inference is to jointly assign values to the parts of the structure, which are represented by a collection of inference variables $\mathbf{y} \in \{0,1\}^n$. For all vectors, subscripts represent their i^{th} component.

Each y_i is associated with a real valued $c_{\mathbf{p},i} \in \Re$ which is the score for the variable y_i being assigned the value 1. We denote the vector comprising of all the $c_{\mathbf{p},i}$ as $\mathbf{c_p}$. The search space for assignments is restricted via constraints, which can be written as a collection of linear inequalities, $\mathbf{M}^T\mathbf{y} \leq \mathbf{b}$. For a problem $\mathbf{p}$, we denote this feasible set of structures by $K_\mathbf{p}$.

The inference problem is that of finding the feasible assignment to the structure which maximizes the dot product $\mathbf{c}^T\mathbf{y}$. Thus, the prediction problem can be written as

$$\arg\max_{\mathbf{y} \in K_\mathbf{p}} \mathbf{c}^T\mathbf{y}. \quad (1)$$

We denote the solution of this maximization problem as $\mathbf{y_p}$.

Let the set $P = \{\mathbf{p}^1, \mathbf{p}^2, \cdots\}$ denote previously solved inference problems, along with their respective solutions $\{\mathbf{y}_\mathbf{p}^1, \mathbf{y}_\mathbf{p}^2, \cdots\}$. An *equivalence class* of integer linear programs, denoted by $[P]$, consists of ILPs which have the same number of inference variables and the same feasible set. Let $K_{[P]}$ denote the feasible set of an equivalence class $[P]$. For a program $\mathbf{p}$, the notation $\mathbf{p} \sim [P]$ indicates that it belongs to the equivalence class $[P]$.

(Srikumar et al., 2012) introduced a set of amortized inference schemes, each of which provides a condition for a new ILP to have the same solution as a previously seen problem. We will briefly review one exact inference scheme introduced in that work. Suppose $\mathbf{q}$ belongs to the same equivalence class of ILPs as $\mathbf{p}$. Then the solution to $\mathbf{q}$ will be the same as that of $\mathbf{p}$ if the following condition holds for all inference variables:

$$(2\mathbf{y}_{\mathbf{p},i} - 1)(\mathbf{c}_{\mathbf{q},i} - \mathbf{c}_{\mathbf{p},i}) \geq 0. \tag{2}$$

This condition, referred to as *Theorem 1* in that work, is the baseline for our experiments.

In general, for any amortization scheme A, we can define two primitive operators TESTCONDITION$_A$ and SOLUTION$_A$. Given a collection of previously solved problems P and a new inference problem $\mathbf{q}$, TESTCONDITION$_A(P, \mathbf{q})$ checks if the solution of the new problem is the same as that of some previously solved one and if so, SOLUTION$_A(P, \mathbf{q})$ returns the solution.

3 Margin-based Amortization

In this section, we will introduce a new method for amortizing inference costs over time. The key observation that leads to this theorem stems from the *structured margin* δ for an inference problem $\mathbf{p} \sim [P]$, which is defined as follows:

$$\delta = \min_{\mathbf{y} \in K_{[P]}, \mathbf{y} \neq \mathbf{y_p}} \mathbf{c}_\mathbf{p}^T(\mathbf{y_p} - \mathbf{y}). \tag{3}$$

That is, for all feasible $\mathbf{y}$, we have $\mathbf{c}_\mathbf{p}^T\mathbf{y_p} \geq \mathbf{c}_\mathbf{p}^T\mathbf{y} + \delta$. The margin δ is the upper limit on the *change in objective* that is allowed for the constraint set $K_{[P]}$ for which the solution will not change.

For a new inference problem $\mathbf{q} \sim [P]$, we define Δ as the maximum change in objective value that can be effected by an assignment that is *not* the solution. That is,

$$\Delta = \max_{\mathbf{y} \in K_{[P]}, \mathbf{y} \neq \mathbf{y_p}} (\mathbf{c_q} - \mathbf{c_p})^T \mathbf{y} \tag{4}$$

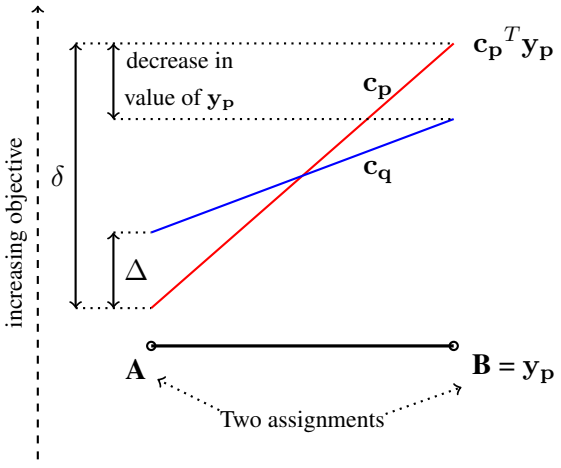

Figure 2: An illustration of the margin-based amortization scheme showing the very simple case with only two competing assignments **A** and **B**. Suppose **B** is the solution $\mathbf{y_p}$ for the inference problem $\mathbf{p}$ with coefficients $\mathbf{c_p}$, denoted by the red hyperplane, and **A** is the second-best assignment. For a new coefficient vector $\mathbf{c_q}$, if the margin δ is greater than the sum of the decrease in the objective value of $\mathbf{y_p}$ and the maximum increase in the objective of another solution (Δ), then the solution to the new inference problem will still be $\mathbf{y_p}$. The margin-based amortization theorem captures this intuition.

Before stating the theorem, we will provide an intuitive explanation for it. Moving from $\mathbf{c_p}$ to $\mathbf{c_q}$, consider the sum of the decrease in the value of the objective for the solution $\mathbf{y_p}$ and Δ, the maximum change in objective value for an assignment that is not the solution. If this sum is less than the margin δ, then no other solution will have an objective value higher than $\mathbf{y_p}$. Figure 2 illustrates this using a simple example where there are only two competing solutions.

This intuition is captured by our main theorem which provides a condition for problems $\mathbf{p}$ and $\mathbf{q}$ to have the same solution $\mathbf{y_p}$.

Theorem 1 (Margin-based Amortization)**.** *Let* $\mathbf{p}$ *denote an inference problem posed as an integer linear program belonging to an equivalence class* $[P]$ *with optimal solution* $\mathbf{y_p}$*. Let* $\mathbf{p}$ *have a structured margin* δ*, i.e., for any* $\mathbf{y}$*, we have* $\mathbf{c}_\mathbf{p}^T\mathbf{y_p} \geq \mathbf{c}_\mathbf{p}^T\mathbf{y} + \delta$*. Let* $\mathbf{q} \sim [P]$ *be another inference instance in the same equivalence class and let* Δ *be defined as in Equation 4. Then,* $\mathbf{y_p}$ *is the solution of the problem* $\mathbf{q}$ *if the following holds:*

$$-(\mathbf{c_q} - \mathbf{c_p})^T\mathbf{y_p} + \Delta \leq \delta \tag{5}$$

Proof. For some feasible $\mathbf{y}$, we have

$$\begin{aligned}\mathbf{c_q}^T\mathbf{y_p} - \mathbf{c_q}^T\mathbf{y} &\geq \mathbf{c_q}^T\mathbf{y_p} - \mathbf{c_p}^T\mathbf{y} - \Delta \\ &\geq \mathbf{c_q}^T\mathbf{y_p} - \mathbf{c_p}^T\mathbf{y_p} + \delta - \Delta \\ &\geq 0\end{aligned}$$

The first inequality comes from the definition of Δ in (4) and the second one follows from the definition of δ. The condition of the theorem in (5) gives us the final step. For any feasible $\mathbf{y}$, the objective score assigned to $\mathbf{y_p}$ is greater than the score assigned to $\mathbf{y}$ according to problem $\mathbf{q}$. That is, $\mathbf{y_p}$ is the solution to the new problem. □

The margin-based amortization theorem provides a general, new amortized inference algorithm. Given a new inference problem, we check whether the inequality (5) holds for any previously seen problems in the same equivalence class. If so, we return the cached solution. If no such problem exists, then we make a call to an ILP solver.

Even though the theorem provides a condition for two integer linear programs to have the same solution, checking the validity of the condition requires the computation of Δ, which in itself is another integer linear program. To get around this, we observe that if any constraints in Equation 4 are relaxed, the value of the resulting maximum can only increase. Even with the increased Δ, if the condition of the theorem holds, then the rest of the proof follows and hence the new problem will have the same solution. In other words, we can solve relaxed, tractable variants of the maximization in Equation 4 and still retain the guarantees provided by the theorem. The tradeoff is that, by doing so, the condition of the theorem will apply to fewer examples than theoretically possible. In our experiments, we will define the relaxation for each problem individually and even with the relaxations, the inference algorithm based on the margin-based amortization theorem outperforms all previous amortized inference algorithms.

The condition in inequality (5) is, in fact, a strict generalization of the condition for Theorem 1 in (Srikumar et al., 2012), stated in (2). If the latter condition holds, then we can show that $\Delta \leq 0$ and $(\mathbf{c_q} - \mathbf{c_p})^T\mathbf{y_p} \geq 0$. Since δ is, by definition, non-negative, the margin-based condition is satisfied.

4 Decomposed Amortized Inference

One limitation in previously considered approaches for amortized inference stems from the expectation that the same *full* assignment maximizes the objective score for different inference problems, or equivalently, that the entire structure is repeated multiple times. Even with this assumption, we observe a speedup in prediction.

However, intuitively, even if entire structures are not repeated, we expect parts of the assignment to be the same across different instances. In this section, we address the following question: *Can we take advantage of the redundancy in components of structures to extend amortization techniques to cases where the full structured output is not repeated?* By doing so, we can store partial computation for future inference problems.

For example, consider the task of part of speech tagging. While the likelihood of two long sentences having the same part of speech tag sequence is not high, it is much more likely that shorter sections of the sentences will share the same tag sequence. We see from Figure 1 that the number of possible structures for shorter sentences is much smaller than the number of sentences. This implies that many shorter sentences share the same structure, thus improving the performance of an amortized inference scheme for such inputs. The goal of decomposed amortized inference is to extend this improvement to larger problems by increasing the size of equivalence classes.

To decompose an inference problem, we use the approach of Lagrangian Relaxation (Lemaréchal, 2001) that has been used successfully for various NLP tasks (Chang and Collins, 2011; Rush and Collins, 2011). We will briefly review the underlying idea[1]. The goal is to solve an integer linear program $\mathbf{q}$, which is defined as

$$\mathbf{q} : \max_{\mathbf{M}^T\mathbf{y} \leq \mathbf{b}} \mathbf{c_q}^T\mathbf{y}$$

We partition the constraints into two sets, say C_1 denoting $\mathbf{M_1}^T\mathbf{y} \leq \mathbf{b_1}$ and C_2, denoting constraints $\mathbf{M_2}^T\mathbf{y} \leq \mathbf{b_2}$. The assumption is that in the absence the constraints C_2, the inference problem becomes computationally easier to solve. In other words, we can assume the existence of a subroutine that can efficiently compute the solution of the relaxed problem $\mathbf{q}'$:

$$\mathbf{q}' : \max_{\mathbf{M_1}^T\mathbf{y} \leq \mathbf{b_1}} \mathbf{c_q}^T\mathbf{y}$$

[1]For simplicity, we only write inequality constraints in the paper. However, all the results here are easily extensible to equality constraints by removing the non-negativity constraints from the corresponding dual variables.

We define Lagrange multipliers $\Lambda \geq \mathbf{0}$, with one λ_i for each constraint in C_2. For problem $\mathbf{q}$, we can define the Lagrangian as

$$L(\mathbf{y}, \Lambda) = \mathbf{c_q}^T\mathbf{y} - \Lambda^T \left(\mathbf{M_2}^T\mathbf{y} - \mathbf{b_2}\right)$$

Here, the domain of $\mathbf{y}$ is specified by the constraint set C_1. The dual objective is

$$\begin{aligned} L(\Lambda) &= \max_{\mathbf{M_1}^T\mathbf{y} \leq \mathbf{b_1}} \mathbf{c_q}^T\mathbf{y} - \Lambda^T \left(\mathbf{M_2}^T\mathbf{y} - \mathbf{b_2}\right) \\ &= \max_{\mathbf{M_1}^T\mathbf{y} \leq \mathbf{b_1}} \left(\mathbf{c_q} - \Lambda^T\mathbf{M_2}\right)^T \mathbf{y} + \Lambda^T\mathbf{b_2}. \end{aligned}$$

Note that the maximization in the definition of the dual objective has the same functional form as $\mathbf{q}'$ and any approach to solve $\mathbf{q}'$ can be used here to find the dual objective $L(\Lambda)$. The dual of the problem $\mathbf{q}$, given by $\min_{\Lambda \geq \mathbf{0}} L(\Lambda)$, can be solved using subgradient descent over the dual variables.

Relaxing the constraints C_2 to define the problem $\mathbf{q}'$ has several effects. First, it can make the resulting inference problem $\mathbf{q}'$ easier to solve. More importantly, removing constraints can also lead to the merging of multiple equivalence classes, leading to fewer, more populous equivalence classes. Finally, removing constraints can decompose the inference problem $\mathbf{q}'$ into smaller independent sub-problems $\{\mathbf{q}^1, \mathbf{q}^2, \cdots\}$ such that no constraint that is in C_1 has active variables from two different sets in the partition.

For the sub-problem $\mathbf{q}^i$ comprising of variables $\mathbf{y}^i$, let the corresponding objective coefficients be $\mathbf{c_{q^i}}$ and the corresponding sub-matrix of $\mathbf{M_2}$ be $\mathbf{M_2^i}$. Now, we can define the dual-augmented sub-problem as

$$\max_{\mathbf{M_1^i}^T\mathbf{y} \leq \mathbf{b_1^i}} \left(\mathbf{c_{q^i}} - \Lambda^T\mathbf{M_2^i}\right)^T \mathbf{y}^i \qquad (6)$$

Solving all such sub-problems will give us a complete assignment for all the output variables.

We can now define the decomposed amortized inference algorithm (Algorithm 1) that performs sub-gradient descent over the dual variables. The input to the algorithm is a collection of previously solved problems with their solutions, a new inference problem $\mathbf{q}$ and an amortized inference scheme A (such as the margin-based amortization scheme). In addition, for the task at hand, we first need to identify the set of constraints C_2 that can be introduced via the Lagrangian.

First, we check if the solution can be obtained without decomposition (lines 1–2). Otherwise,

Algorithm 1 Decomposed Amortized Inference

Input: A collection of previously solved inference problems P, a new problem $\mathbf{q}$, an amortized inference algorithm A.
Output: The solution to problem $\mathbf{q}$

1: **if** TESTCONDITION(A, $\mathbf{q}$, P) **then**
2: **return** SOLUTION(A, $\mathbf{q}$, P)
3: **else**
4: Initialize $\lambda_i \leftarrow 0$ for each constraint in C_2.
5: **for** $t = 1 \cdots T$ **do**
6: Partition the problem $\mathbf{q}$ into sub-problems $\mathbf{q}^1, \mathbf{q}^2, \cdots$ such that no constraint in C_1 has active variables from two partitions.
7: **for** partition $\mathbf{q}^i$ **do**
8: $\mathbf{y}^i \leftarrow$ Solve the maximization problem for $\mathbf{q}^i$ (Eq. 6) using the amortized scheme A.
9: **end for**
10: Let $\mathbf{y} \leftarrow \left[\mathbf{y}^1; \mathbf{y}^2; \cdots\right]$
11: **if** $\mathbf{M_2 y} \leq \mathbf{b_2}$ and $(\mathbf{b_2} - \mathbf{M_2 y})_i \lambda_i = 0$ **then**
12: **return** $\mathbf{y}$
13: **else**
14: $\Lambda \leftarrow \left[\Lambda - \mu_t \left(\mathbf{b_2} - \mathbf{M_2}^T\mathbf{y}\right)\right]_+$
15: **end if**
16: **end for**
17: **return** solution of $\mathbf{q}$ using a standard inference algorithm
18: **end if**

we initialize the dual variables Λ and try to obtain the solution iteratively. At the t^{th} iteration, we partition the problem $\mathbf{q}$ into sub-problems $\{\mathbf{q}^1, \mathbf{q}^2, \cdots\}$ as described earlier (line 6). Each partition defines a smaller inference problem with its own objective coefficients and constraints. We can apply the amortization scheme A to each sub-problem to obtain a complete solution for the relaxed problem (lines 7–10). If this solution satisfies the constraints C_2 and complementary slackness conditions, then the solution is provably the maximum of the problem $\mathbf{q}$. Otherwise, we take a subgradient step to update the value of Λ using a step-size μ_t, subject to the constraint that all dual variables must be non-negative (line 14). If we do not converge to a solution in T iterations, we call the underlying solver on the full problem.

In line 8 of the algorithm, we make multiple calls to the underlying amortized inference procedure to solve each sub-problem. If the sub-

problem cannot be solved using the procedure, then we can either solve the sub-problem using a different approach (effectively giving us the standard Lagrangian relaxation algorithm for inference), or we can treat the full instance as a cache miss and make a call to an ILP solver. In our experiments, we choose the latter strategy.

5 Experiments and Results

Our experiments show two results: 1. The margin-based scheme outperforms the amortized inference approaches from (Srikumar et al., 2012). 2. Decomposed amortized inference gives further gains in terms of re-using previous solutions.

5.1 Tasks

We report the performance of inference on two NLP tasks: semantic role labeling and the task of extracting entities and relations from text. In both cases, we used an existing formulation for structured inference and only modified the inference calls. We will briefly describe the problems and the implementation and point the reader to the literature for further details.

Semantic Role Labeling (SRL) Our first task is that of identifying arguments of verbs in a sentence and annotating them with semantic roles (Gildea and Jurafsky, 2002; Palmer et al., 2010) . For example, in the sentence *Mrs. Haag plays Eltiani.*, the verb *plays* takes two arguments: *Mrs. Haag*, the actor, labeled as A0 and *Eltiani*, the role, labeled as A1. It has been shown in prior work (Punyakanok et al., 2008; Toutanova et al., 2008) that making a globally coherent prediction boosts performance of SRL.

In this work, we used the SRL system of (Punyakanok et al., 2008), where one inference problem is generated for each verb and each inference variables encodes the decision that a given constituent in the sentence takes a specific role. The scores for the inference variables are obtained from a classifier trained on the PropBank corpus. Constraints encode structural and linguistic knowledge about the problem. For details about the formulations of the inference problem, please see (Punyakanok et al., 2008).

Recall from Section 3 that we need to define a relaxed version of the inference problem to efficiently compute Δ for the margin-based approach. For a problem instance with coefficients $\mathbf{c_q}$ and cached coefficients $\mathbf{c_p}$, we take the sum of the highest n values of $\mathbf{c_q} - \mathbf{c_p}$ as our Δ, where n is the number of argument candidates to be labeled.

To identify constraints that can be relaxed for the decomposed algorithm, we observe that most constraints are not predicate specific and apply for all predicates. The only constraint that is predicate specific requires that each predicate can only accept roles from a list of roles that is defined for that predicate. By relaxing this constraint in the decomposed algorithm, we effectively merge all the equivalence classes for all predicates with a specific number of argument candidates.

Entity-Relation extraction Our second task is that of identifying the types of entities in a sentence and the relations among them, which has been studied by (Roth and Yih, 2007; Kate and Mooney, 2010) and others. For the sentence *Oswald killed Kennedy*, the words *Oswald* and *Kennedy* will be labeled by the type PERSON, and the KILL relation exists between them.

We followed the experimental setup as described in (Roth and Yih, 2007). We defined one inference problem for each sentence. For every entity (which is identified by a constituent in the sentence), an inference variable is introduced for each entity type. For each pair of constituents, an inference variable is introduced for each relation type. Clearly, the assignment of types to entities and relations are not independent. For example, an entity of type ORGANIZATION cannot participate in a relation of type BORN-IN because this relation label can connect entities of type PERSON and LOCATION only. Incorporating these natural constraints during inference were shown to improve performance significantly in (Roth and Yih, 2007). We trained independent classifiers for entities and relations and framed the inference problem as in (Roth and Yih, 2007). For further details, we refer the reader to that paper.

To compute the value of Δ for the margin-based algorithm, for a new instance with coefficients $\mathbf{c_q}$ and cached coefficients $\mathbf{c_p}$, we define Δ to be the sum of all non-negative values of $\mathbf{c_q} - \mathbf{c_p}$.

For the decomposed inference algorithm, if the number of entities is less than 5, no decomposition is performed. Otherwise, the entities are partitioned into two sets: set A includes the first four entities and set B includes the rest of the entities. We relaxed the relation constraints that go across these two sets of entities to obtain two independent inference problems.

5.2 Experimental Setup

We follow the experimental setup of (Srikumar et al., 2012) and simulate a long-running NLP process by caching problems and solutions from the Gigaword corpus. We used a database engine to cache ILP and their solutions along with identifiers for the equivalence class and the value of δ.

For the margin-based algorithm and the Theorem 1 from (Srikumar et al., 2012), for a new inference problem $\mathbf{p} \sim [P]$, we retrieve all inference problems from the database that belong to the same equivalence class $[P]$ as the test problem $\mathbf{p}$ and find the cached assignment $\mathbf{y}$ that has the highest score according to the coefficients of $\mathbf{p}$. We only consider cached ILPs whose solution is $\mathbf{y}$ for checking the conditions of the theorem. This optimization ensures that we only process a small number of cached coefficient vectors.

In a second efficiency optimization, we pruned the database to remove redundant inference problems. A problem is redundant if solution to that problem can be inferred from the other problems stored in the database that have the same solution and belong to the same equivalence class. However, this pruning can be computationally expensive if the number of problems with the same solution and the same equivalence class is very large. In that case, we first sampled a 5000 problems randomly and selected the non-redundant problems from this set to keep in the database.

5.3 Results

We compare our approach to a state-of-the-art ILP solver[2] and also to Theorem 1 from (Srikumar et al., 2012). We choose this baseline because it is shown to give the highest improvement in wall-clock time and also in terms of the number of cache hits. However, we note that the results presented in our work outperform all the previous amortization algorithms, including the approximate inference methods.

We report two performance metrics – the percentage decrease in the number of ILP calls, and the percentage decrease in the wall-clock inference time. These are comparable to the *speedup* and *clock speedup* defined in (Srikumar et al., 2012). For measuring time, since other aspects of prediction (like feature extraction) are the same across all settings, we only measure the time taken for inference and ignore other aspects. For both tasks, we report the runtime performance on section 23 of the Penn Treebank. Note that our amortization schemes *guarantee optimal solution*. Consequently, using amortization, task accuracy remains the same as using the original solver.

Table 1 shows the percentage reduction in the number of calls to the ILP solver. Note that for both the SRL and entity-relation problems, the margin-based approach, even without using decomposition (the columns labeled **Original**), outperforms the previous work. Applying the decomposed inference algorithm improves both the baseline and the margin-based approach. Overall, however, the fewest number of calls to the solver is made when combining the decomposed inference algorithm with the margin-based scheme. For the semantic role labeling task, we need to call the solver only for one in six examples while for the entity-relations task, only one in four examples require a solver call.

Table 2 shows the corresponding reduction in the wall-clock time for the various settings. We see that once again, the margin based approach outperforms the baseline. While the decomposed inference algorithm improves running time for SRL, it leads to a slight increase for the entity-relation problem. Since this increase occurs in spite of a reduction in the number of solver calls, we believe that this aspect can be further improved with an efficient implementation of the decomposed inference algorithm.

6 Discussion

Lagrangian Relaxation in the literature In the literature, in applications of the Lagrangian relaxation technique (such as (Rush and Collins, 2011; Chang and Collins, 2011; Reichart and Barzilay, 2012) and others), the relaxed problems are solved using specialized algorithms. However, in both the relaxations considered in this paper, even the relaxed problems cannot be solved without an ILP solver, and yet we can see improvements from decomposition in Table 1.

To study the impact of amortization on running time, we modified our decomposition based inference algorithm to solve each sub-problem using the ILP solver instead of amortization. In these experiments, we ran Lagrangian relaxation for until convergence or at most T iterations. After T iterations, we call the ILP solver and solve the original problem. We set T to 100 in one set of exper-

[2] We used the Gurobi optimizer for our experiments.

Method	% ILP Solver calls required			
	Semantic Role Labeling		Entity-Relation Extraction	
	Original	+ Decomp.	Original	+ Decomp.
ILP Solver	100	–	100	–
(Srikumar et al., 2012)	41	24.4	59.5	57.0
Margin-based	32.7	16.6	28.2	25.4

Table 1: Reduction in number of inference calls

Method	% time required compared to ILP Solver			
	Semantic Role Labeling		Entity-Relation Extraction	
	Original	+ Decomp.	Original	+ Decomp.
ILP Solver	100	–	100	–
(Srikumar et al., 2012)	54.8	40.0	81	86
Margin-based	45.9	38.1	58.1	61.3

Table 2: Reduction in inference time

iments (call it $Lag1$) and T to 1 (call it $Lag2$). In SRL, compared to solving the original problem with ILP Solver, both $Lag1$ and $Lag2$ are roughly 2 times slower. For entity relation task, compared to ILP Solver, $Lag1$ is 186 times slower and $Lag2$ is 1.91 times slower. Since we used the same implementation of the decomposition in all experiments, this shows that the decomposed inference algorithm crucially benefits from the underlying amortization scheme.

Decomposed amortized inference The decomposed amortized inference algorithm helps improve amortized inference in two ways. First, since the number of structures is a function of its size, considering smaller sub-structures will allow us to cache inference problems that cover a larger subset of the space of possible sub-structures. We observed this effect in the problem of extracting entities and relations in text. Second, removing a constraint need not always partition the structure into a set of smaller structures. Instead, by removing the constraint, examples that might have otherwise been in different equivalence classes become part of a combined, larger equivalence class. Increasing the size of the equivalence classes increases the probability of a cache-hit. In our experiments, we observed this effect in the SRL task.

7 Conclusion

Amortized inference takes advantage of the regularities in structured output to re-use previous computation and improve running time over the lifetime of a structured output predictor. In this paper, we have described two approaches for amortizing inference costs over datasets. The first, called the *margin-based amortized inference*, is a new, provably exact inference algorithm that uses the notion of a structured margin to identify previously solved problems whose solutions can be reused. The second, called *decomposed amortized inference*, is a meta-algorithm over any amortized inference that takes advantage of previously computed sub-structures to provide further reductions in the number of inference calls. We show via experiments that these methods individually give a reduction in the number of calls made to an inference engine for semantic role labeling and entity-relation extraction. Furthermore, these approaches complement each other and, together give an additional significant improvement.

Acknowledgments

The authors thank the members of the Cognitive Computation Group at the University of Illinois for insightful discussions and the anonymous reviewers for valuable feedback. This research is sponsored by the Army Research Laboratory (ARL) under agreement W911NF-09-2-0053. The authors also gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. This material also is based on research sponsored by DARPA under agreement number FA8750-13-2-0008. This work has also been supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20155. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of ARL, DARPA, AFRL, IARPA, DoI/NBC or the US government.

References

Y-W. Chang and M. Collins. 2011. Exact decoding of phrase-based translation models through Lagrangian relaxation. EMNLP.

J. Clarke and M. Lapata. 2006. Constraint-based sentence compression: An integer programming approach. In *ACL*.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*.

R. Kate and R. Mooney. 2010. Joint entity and relation extraction using card-pyramid parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 203–212. Association for Computational Linguistics.

T. Koo, A. M. Rush, M. Collins, T. Jaakkola, and D. Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *EMNLP*.

C. Lemaréchal. 2001. Lagrangian Relaxation. In *Computational Combinatorial Optimization*, pages 112–156.

M. Palmer, D. Gildea, and N. Xue. 2010. *Semantic Role Labeling*, volume 3. Morgan & Claypool Publishers.

V. Punyakanok, D. Roth, and W. Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*.

R. Reichart and R. Barzilay. 2012. Multi event extraction guided by global constraints. In *NAACL*, pages 70–79.

S. Riedel and J. Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *EMNLP*.

S. Riedel. 2009. Cutting plane MAP inference for Markov logic. *Machine Learning*.

D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *CoNLL*.

D. Roth and W. Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*.

A.M. Rush and M. Collins. 2011. Exact decoding of syntactic translation models through Lagrangian relaxation. In *ACL*, pages 72–82, Portland, Oregon, USA, June.

A. M. Rush, D. Sontag, M. Collins, and T. Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *EMNLP*.

V. Srikumar, G. Kundu, and D. Roth. 2012. On amortizing inference cost for structured prediction. In *EMNLP*.

K. Toutanova, A. Haghighi, and C. D. Manning. 2008. A global joint model for semantic role labeling. *Computational Linguistics*, 34:161–191.