Sparse Belief Propagation for Joint Inference of Entities, Relations, and Coreference

Abstract

Although joint inference has been a popular approach to avoid cascading of errors when inferring multiple natural language tasks, its application to information extraction has been limited to modeling only two tasks at a time. In this paper, we focus on three crucial tasks in the information extraction pipeline: entity tagging, relation extraction, and coreference. We propose a single, joint graphical model that represents the various dependencies between the tasks, allowing bi-directional flow of uncertainty between the tasks. Since the resulting model has a high tree-width and contains a large number of variables, we present a novel extension to belief propagation that sparsifies the domains of variables during inference. Experimental results show that our joint model provides consistently improved results on all three tasks as more dependencies are captured, in particular our joint model obtains 12% error reduction on tagging over the isolated model, and our inference technique is 30 times faster than regular belief propagation.

1 Introduction

Most natural language processing tasks are decomposed into a number of subtasks, for example chunking, named entity recognition, parsing, semantic role labeling and part-of-speech tagging. For inference, independently-trained, accurate models for each of these tasks are placed in a pipeline system, with the output prediction of these models feeding into downstream modules as input. These pipeline systems, however, are restricted to a uni-directional flow of information, and suffer from cascading errors. To address these concerns, there has been some past and growing recent interest in joint inference of multiple NLP tasks (McCallum et al., 1999; Finkel et al., 2006; Poon and Domingos, 2007) that allows bi-directional information flow and reduces the cascading of errors.

Much of the previous work on joint inference has focused on information extraction. For example, Finkel and Manning (2009) perform named-entity recognition and parsing jointly, obtaining improvement on both tasks. Roth and Yih (2007) perform joint inference for entity and relation identification and classification. Haghighi and Klein (2010) propose a generative model that jointly predicts entity types and coreference decisions. These approaches demonstrate the benefits and importance of joint inference by improving the accuracy on the individual information extraction subtasks.

These methods, however, are limited by the tasks they consider for joint inference. Instead of representing the whole information extraction system as a single model, these approaches only model two tasks jointly and continue to suffer from cascading errors due to the remaining tasks. Further, coreference is an important module, the prediction of which can improve all other components of the pipeline. Unfortunately, most of these systems do not consider coreference jointly with other tasks, and the ones that do use sampling-based inference to incorporate it, an approach that suffers from local minima and requires a customized, problem-specific proposal function.

In this paper, we propose a single, joint prob-
Schumacher has a contract with the Italian team through 2002.

Figure 1: Information Extraction: An example of a sentence labeled with 3 mentions labeled with entity types (red boxes), relations (green arrows), and coreference (shown as blue links to three other sentences from the document).

tablistic graphical model for classification of text mentions (entity tagging), clustering of mentions that refer to the same entity (coreference resolution), and identification of the relations between these entities (relation extraction). Figure 1 shows an example annotated sentence from ACE 2004 (Doddington et al., 2004), a standard newswire dataset. We expect our joint model to reduce the cascading error by facilitating bi-directional information flow, allowing entity tags to be improved from better relation extraction (since certain relations, such as EMPLOY can only occur between a PERSON and an ORGANIZATION) and from coreference resolution (as mentions which are coreferent have an identical entity type). To deal with our joint model’s high treewidth and large number of variables and factors, we introduce a modification to belief propagation that facilitates efficient inference. In particular, during the course of inference we examine the marginals of individual variables and fix the variables to an observed value if the entropy of the marginal becomes low. This value sparsification allows us to remove a large number of factors, efficiently marginalize over the remaining factors, and decompose the model into pieces for which inference may be done in parallel.

Our experimental results show that joint inference provides higher accuracy for all three tasks. In spite of the obvious benefits of joint inference, previous work has faced difficulty in obtaining high accuracy with joint inference. For the task of joint relation and entity labeling, Roth and Yih (2007) show negative improvements on entity labeling. Sutton and McCallum (2005a) present a joint model of parsing and semantic role labeling that performs worse than the pipeline approach. In the CoNLL-2008 shared task on joint parsing and semantic role labeling (Surdeanu et al., 2008) the top five systems in the closed challenge consisted of pipeline approaches. Even when positive results are demonstrated, as in Finkel and Manning (2009) and Kate and Mooney (2010), they are modest. Comparatively, we achieve an entity tagging error reduction of 12.4% for the joint model. Further, our results show consistent improvement as more tasks are included in the model, demonstrating the utility of joint inference.

2 Isolated Models

In this section, we present a brief background on probabilistic graphical models, before describing the three tasks of our interest, and laying out the model and the features commonly used to represent them.

2.1 Graphical Models

Graphical models define a family of probability distributions that factorize according to the dependencies encoded in the graph structure. Factor graphs are commonly used to represent undirected graphical models (Kschischang et al., 2001). Formally, a factor graph $G$ is a bipartite graph with variable nodes $y$, and factor nodes $\Psi = \{\psi_i\}$, where $y$ is the variables to represent the distribution over. The probability distribution can be written as $p(y) = \frac{1}{Z} \prod_{\psi_i \in \Psi} \psi_i$, where $Z = \sum_y \prod_{\psi_i} \psi_i$ is the partition function that ensures the probabilities sum to one. Factors are often defined as a log-linear combi-
nation of the inner product of sufficient statistics (or feature functions) $f_k$ and parameters $\lambda_k$. Graphical models are a popular tool in NLP to represent the tasks, and we shall use the above notation in the rest of the text.

### 2.2 Entity Tagging

Entity tagging is the task of classifying each mention according to the type of entity to which they refer. The input for this task are the set observed mention strings and their boundaries. For each mention $m_i$, the output of entity tagging is a label $t_i$ from a pre-defined set of labels $T$. The specific set of labels $T$ that are used depends on the domain of interest, for example the set of labels used in ACE consist of PERSON, ORGANIZATION, GEO-POLITICAL, LOCATION, FACILITY, VEHICLE, and WEAPON.

As an example of entity tagging, consider the sentence “A cap for a depressurization valve floated away moments after Bill McArthur emerged from space shuttle Discovery,” where the correct labels for the two mentions are PERSON and VEHICLE, respectively.

A common approach, and the approach that we follow, is to treat entity recognition as a classification task using a maximum entropy model (Bender et al., 2003). This model may be written as a graphical model by defining a factor $\Psi_T(t_i)$ for each entity tag variable $t_i$. We use several granularities for features: word-level, mention-level, and sentence-level. For word-level features, we have leveraged research on a similar\(^1\) task, Named Entity Recognition (NER). These are the word string, stem, shape, part-of-speech, and whether a digit or punctuation character appears. We also use membership in each of the 30 lexicons from Ratimov and Roth (2009), which cover Wikipedia entities, common names, countries, monetary units, temporal expressions, etc. Mention-level features consist of the mention string, lexicon memberships, bi-grams of mention tokens, and mention type. Additionally, dependency parse features are used: token-level features of the head of the mention, the first token, and the last token and also the stem of the closest verb.

\[^1\]In NER as defined by the CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003) entity mentions spans and labels must be predicted. Here, we are only predicting labels.

<table>
<thead>
<tr>
<th>Types</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Token</td>
<td>String, Stem, Shape, Part-of-speech, Punctuation, Digit, Lexicon memberships</td>
</tr>
<tr>
<td>Mention</td>
<td>String, Lexicon memberships, Bi-grams, Token-level of head, first, and last tokens, verb stem closest to head token according to dependency parse, type</td>
</tr>
<tr>
<td>Sentence</td>
<td>Mention position in sentence, mention length, verb stems, Token-level of token preceding mention</td>
</tr>
</tbody>
</table>

Table 1: Entity tagging features

On the sentence-level, we simply add position features of the mention (measured in tokens from the start and end of sentence) and the token-level features of the token preceding the mention. These features are also listed in Table 1.

### 2.3 Relation Extraction

The relation extraction task is to discover pairs of entity mentions that express a relationship and to tag that relationship with the appropriate relation type. The input to this task consists of observed entity mentions with boundaries from a text corpus. For pairs of mentions that appear in the same sentence, relation extraction must identify the first and the second argument of the relation and tag it with the appropriate relation type (or NONE if no relation is expressed between the pair). This task is often represented as variables $r_{ij}$ that represent the type of the relation where $m_i$ is the first argument, $m_j$ the second argument, and the type comes from a predefined set of labels $R$.\(^2\) The set of relation types $R$ used in ACE is given in Table 2. In the example in Figure 1, there are two relations: EMPLOY-STAFF between “Schumacher” and “the Italian team”, and BASED IN between “the Italian team” and “Italian”.

As with the other tasks, a common model for relation extraction is to independently label each relation mention with its type (Jiang and Zhai, 2007). This model is represented as factor templates $\Psi_R^L(r_{ij})$ and $\Psi_R^K(t_i, t_j)$ over variables.

The features are drawn from related work on relation extraction, (Zhou et al., 2005) in particular. Word features include bag-of-words of each men-

\[^2\]Note that, $r_{ij}$ and $r_{ji}$ are distinct variables.
### Relation Type: Subtype, Subtype...

<table>
<thead>
<tr>
<th>NONE:</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMP-ORG: Employ-Executive, Employ-Staff, Subsidiary, Member-of-Group, Employ-Undetermined, Partner, Other</td>
</tr>
<tr>
<td>GPE-AFF: Based-In, Citizen-or-Resident, Other</td>
</tr>
<tr>
<td>PHYS: Part-Whole, Located, Near</td>
</tr>
<tr>
<td>PER-SOC: Business, Family, Other</td>
</tr>
<tr>
<td>OTHER-AFF: Ideology, Ethnic, Other</td>
</tr>
<tr>
<td>ART: User-or-Owner, Inventor-or-Manufacturer, Other</td>
</tr>
<tr>
<td>DISC: None</td>
</tr>
</tbody>
</table>

Table 2: ACE Relation Types

<table>
<thead>
<tr>
<th>Types</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Words</td>
<td>bag-of-words, head word, conjunction of heads, words between mentions, words preceding and following the mentions</td>
</tr>
<tr>
<td>Mention</td>
<td>conjunction of mention types</td>
</tr>
<tr>
<td>Overlap</td>
<td>number of mentions/words in between, nesting</td>
</tr>
<tr>
<td>Chunking</td>
<td>phrase heads in between, phrase heads preceding and following mentions, path of phrase labels</td>
</tr>
</tbody>
</table>

Table 3: Relation Extraction Features

Coreference is the task of linking mentions within a document which refer to the same real-word entity. The input are mentions of a document, and the system outputs entities, which are clusters of mentions. In the example of Figure 1, “Schumacher,” “Micheal Schumacher,” and “I” all refer to the same person. Thus, they should be linked together as one entity.

A common approach to the coreference task is to classify pairs of mentions as coreferent or not, i.e. for pairs of mentions $m_i$ and $m_j$ that appear in the same document, there is a variable $c_{ij} \in \{0, 1\}$. These decisions are symmetric ($c_{ij} \equiv c_{ji}$), and we only include one of these variables in the model. Coreference also requires the variables to be consistent with the transitive closure. Since including transitivity as factors will require $O(n^3)$ factors, it is not scalable. Instead, we compute transitive closures of the coreferent pairs as a post processing step. We exclude pairs where both mentions are pronouns, as well as pairs that involve one pronoun but the two mentions are more than three sentences apart. As in Bengston and Roth (2008), we do not include pairs whose first mention is pronominal and the second is non-pronominal.

The parameters of the model are defined by a factor template $\Psi_C(c_{ij}, t_i, t_j)$. We used features similar to those used in Soon et al. (2001) and Bengston and Roth (2008). The string similarity features include whether the two mention strings match, whether the head token of the mentions match, whether the prefix or suffix matches. Syntactic features include whether the gender (masculine, feminine, neuter) of the two mentions match, whether the number (singular, plural) matches, the conjunction of the POS tags of the head tokens, whether the two mentions are in an appositive, relative, or predicate nominative structure, the type of the mentions (nominal, pronominal, proper, premodifier). We used WordNet to determine whether the two mentions are synonyms, antonyms or hypernyms. We also used the number of sentences and tokens between the two mentions and capitalization as features. We also include conjunctions of these features. These features are also listed in Table 4.

### 3 Joint Model

We would like a model that directly represents the uncertainty between the three tasks. Because the individual models defined in the previous section represent the individual tasks, we construct a joint model by combining all the variables and factors into a single graphical model. To elaborate, the resulting model has the same variables and features as...
Table 4: Coreference features

<table>
<thead>
<tr>
<th>Types</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Similarities</td>
<td>exact string match, head match, substring match, acronym</td>
</tr>
<tr>
<td>Syntax</td>
<td>number match, gender match, POS tags, apposition, relative pronoun, predicate nominative structure, mention type, NP type (definite, demonstrative, pronoun), grammatical subject and object</td>
</tr>
<tr>
<td>Semantic</td>
<td>synonym, antonym, hypernym</td>
</tr>
<tr>
<td>Other</td>
<td>sentence/token distance, capitalization</td>
</tr>
</tbody>
</table>

the individual models. There exist entity tag variables, relation label variables, and pairwise coreference decision variables. The factors of the joint models are also the set of factors instantiated by individual models, as described in sections 2.2, 2.3, and 2.4. See Figure 2 for an illustration of the joint model as defined over 3 mentions, two of which are in the same sentence. Note that even with such a small set of mentions, the underlying joint model is quite complex and dense. Variable counts over the train, test, and development sets are provided in Table 5.

Formally, the probability for a setting to all the variables in a document is:

\[
p(t, r, c) \propto \prod_{t_i \in t} \Psi_T(t_i) \prod_{c_{ij} \in c} \Psi_C(c_{ij}, t_i, t_j) \prod_{r_{ij} \in r} \Psi_L^T(r_{ij}) \Psi_L^C(r_{ij}, t_i, t_j) \tag{1}
\]

The semantics of these factors are slightly different from earlier. Instead of representing a distribution over the labels of a single task given the predictions from another task, these factors now directly represent the joint probability distribution over the tasks that they are defined over. For example, in Section 2.4, each coreference factor defines a distribution over the pairwise boolean coreference variable given the entity tags of the mentions. In the joint model, however, this factor induces a distribution over both the pairwise boolean variable and the entity tags of the two mentions, based on the observed features between the two mentions. When trained, this factor can capture the bi-directional information flow between the tasks and, for example, encourage the entity tags of two mentions to be the same if confident about them being coreferent. Similarly, the relation extraction factors also induce a distribution over the entity tags of their arguments.

Note that the coreference resolution and relation extraction are not directly connected in the model, as the dependency between these two tasks is much weaker in practice. Nonetheless they are not independent. As part of the same graphical model, information can flow between the two via entity tags, resulting in indirect improvements to relation extraction when coreference improves.

4 Learning and Inference

Given the large size and complex structure of the joint model, none of the existing approaches to inference and learning can be directly applied. Instead we propose a novel extension to the belief propagation algorithm that allows inference to scale, along with estimating parameters using modifications to an existing learning approach.

4.1 Piecewise Learning for Joint Models

The objective of learning is to identify the set of parameters that maximize the likelihood of the la-
beled data, which, for our joint model, will be the joint likelihood over all the three tasks. Common approaches to maximize the training objective, such as BFGS, unfortunately cannot be applied to our setting due to a number of reasons. First, as we will outline below, exact inference is NP-Hard, and even common approximate inference techniques are computationally expensive. Learning with approximate inference for such models can often converge (Kulesza and Pereira, 2008). Second, the likelihood term is defined over all the tasks simultaneously, and the optimization approach faces difficulty balancing between the different tasks, often biasing the learning for the task with most terms in the objective, which, in our case, is coreference resolution. Third, since the factors was defined over a much larger domain (joint distribution over multiple domains), the number of parameters to optimize can be really large (often resulting in billions, if not explicitly restricted).

Our approach to learning attempts to address these concerns. Because joint training is intractable due to the complexity of inference, we use the piecewise training (Sutton and McCallum, 2009) approach to learn our models. Instead of learning all the factors jointly, this approach decomposes the model into pieces, and maximizes the piecewise likelihood by treating each piece independently. For our joint model, we treat each factor as an independent piece, separately learning the distribution over its neighbors given the observed features. Further, to facilitate faster convergence, the predictions from entity tagging factors are incorporated during piecewise training of relation extraction and coreference as fixed incoming beliefs. To limit the number of parameters that occur in the joint model between entity tagging and relation extraction, we only include the supported features, i.e., the features that appear at least once in the training data. These modifications to existing approaches enable tractable parameter estimation of the joint model.

4.2 Efficient Inference with Sparsity

Due to the number of variables, non-trivial domain sizes, strong dependencies, and a loopy structure, common approximate inference techniques, such as belief propagation and sampling, cannot be applied directly. Belief propagation converges to accurate marginals in a few iterations when the model is, for the most part, cycle-free, and when marginalization of each factor is quick. Unfortunately, as described in Section 3, the joint model is incredibly loopy due to the large number of coreference variables that connect variables over the whole document. Further, even marginalization of individual factors is non-trivial due to the large domain involved (for example, the neighborhood of $\Psi_p$ consists of all possible combinations of the relation label along with the entity tags for the pair of argument entities). These reasons prevent the use of belief propagation in our model. An alternative approach that addresses a number of these concerns is MCMC sampling. Although its performance is robust to the loopy structure and large domains, sampling-based approaches are vulnerable to the local minima problem when faced with strong dependencies between tasks. Hand-built, customized proposal distributions need to be designed to escape these minima (Singh et al., 2009), which we would like to avoid.

Belief propagation provides a number of benefits for joint inference such as fast convergence and robust handling of strong dependencies, and we adapt the algorithm for inference on our model. Our main extension stems from the insight that during inference in NLP models, most of the variable marginals often peak during the initial stages of inference, without changing substantially during the rest of the course of inference. Detecting these low-entropy marginals in earlier phases, and fixing to their high-probability values, can provide a number of benefits to belief propagation. First, since the domain now contains only a single value, the factors that neighbor the variable can marginalize more efficiently (for example, fixing an entity tag helps marginalization of all the $O(n)$ coreference factors that touch it). Second, these fixed variables allow decomposition of the models into independent inference problems by partitioning at these fixed variables (facilitating multi-core parallelization). Lastly, factors that only neighbor fixed variables can be effectively removed during inference, reducing the amount of messages to pass.

To employ these benefits of value sparsity in belief propagation, we examine the marginals of all the variables after every iteration of message passing. When the probability of a value for a variable goes
above a predetermined probability threshold \( \zeta \), we set the value of the variable to its maximum probability value, treating it as a fixed variable for the rest of inference. The parameter \( \zeta \) directly controls the computational efficiency and accuracy trade-off, and we set the value for this parameter based on observing inference on the held out training data.

Since convergence of belief propagation is sensitive to the order of messages, we avoid performing completely random loopy belief propagation. Instead, we initialize the messages by propagating the entity tag factor \( \Psi_T \), followed by propagation of the messages along the joint factors (\( \Psi^L_R \), \( \Psi^J_R \) and \( \Psi^C \)) for a few iterations. We incorporate transitivity into the inference technique by directly propagating the sparse coreference decisions, ignoring the coreference variables that are not fixed. We perform a few iterations of message passing using the above schedule, although we find in practice that iterations subsequent to the first two offer negligible benefits.

To handle transitivity, we only propagate positive sparse coreferent decisions, while being consistent with negative sparse decisions. During the final iteration, we fix all the values of the entity tags to their max-marginal tags, followed by prediction of relation extraction and coreference using the standard isolated models.

5 Related Work

5.1 Individual Tasks

There has been considerable research on the individual tasks covered in this paper. The features in our model are based on these.

Relation extraction systems generally fall into two categories. Feature-based systems (Zhou et al., 2005; Sun et al., 2011) employ a variety of features, including lexical, syntactical and semantic ones. The other common approach is computing similarity between trees using the convolution tree kernel (Zhang et al., 2006). Zhou et al. (2007) proposed a composite of the tree kernel and a linear kernel that outperformed the individual kernels. Jiang and Zhai (2007) systematically explored the feature space and showed that using more than the basic features only yields small improvements. Nguyen and Moschitti (2011) built a composite kernel similar to (Zhou et al., 2007) and found that distant supervision using Wikipedia data improved performance. Hoffmann et al. (2011) considered cases when multiple relations hold on the same entity mentions.

A majority of systems cast the coreference resolution task as binary classifications (Bengston and Roth, 2008). Haghighi and Klein (2009) designed a deterministic system based on a rich set of syntactic and semantic compatibilities. On a high level, their method is similar to ours, except that the pairwise decisions are not predicted by a statistical learner, but by deterministic rules. Unlike our model that computes the transitive closure from all the pairwise predictions, Soon et al. (2001) used the most recent positive antecedent, and Ng and Cardie (2002) linked to the best antecedent among all candidates for each mention. Culotta et al. (2007) argued that the mention-pair method cannot capture features of sets of noun phrases. They constructed features over a cluster of NPs using first-order logic, and augmented the pairwise model to rank sets of NPs.

5.2 Joint Tasks

In recent years, there has been an increasing interest in approaches to joint representations of multiple information extraction and natural language processing tasks (McCallum and Jensen, 2003). Most relevant to our work is the combination of entity labeling and relation extraction. Roth and Yih (2007) use an Integer Linear Program to enforce consistency. Kate and Mooney (2010) use Card-Pyramid parsing, an algorithm similar to CFG parser. Yao et al. (2010) accomplish this through distant supervision via Wikipedia.

Others have combined parsing with NER (Finkel and Manning, 2009) and semantic role labeling (Sutton and McCallum, 2005b) with mixed success. Finkel et al. (2006) represents a pipeline of NLP tasks as a Bayesian network where each variable represents one stage of the pipeline. Joint inference has also been applied to various information extraction tasks such as citation segmentation and matching (Wellner et al., 2004; Poon and Domingos, 2007; Singh et al., 2009) and to BioNLP (Poon and Vanderwende, 2010; Riedel and McCallum, 2011).

Our approach to joint inference differs significantly to these. First, we are modeling three crucial information extraction tasks, including coreference, which have not been modeled together before. Sec-
Table 5: Number of variables of each type in ACE train, test, and development sets.

<table>
<thead>
<tr>
<th>Source</th>
<th>#Ment.</th>
<th>#Coref.</th>
<th>#Relat.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>15,640</td>
<td>637,160</td>
<td>82,479</td>
</tr>
<tr>
<td>Test</td>
<td>6,598</td>
<td>342,942</td>
<td>38,270</td>
</tr>
<tr>
<td>Dev</td>
<td>5,545</td>
<td>244,461</td>
<td>34,057</td>
</tr>
</tbody>
</table>

Table 7: Entity Tagging Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Error Red.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isolated Model</td>
<td>80.23</td>
<td>-</td>
</tr>
<tr>
<td>Joint w/ Coreference</td>
<td>81.24</td>
<td>5.1</td>
</tr>
<tr>
<td>Joint w/ Relations</td>
<td>81.77</td>
<td>7.8</td>
</tr>
<tr>
<td>Complete Joint</td>
<td>82.69</td>
<td>12.4</td>
</tr>
</tbody>
</table>

Table 8: Relation Extraction Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Prec</th>
<th>Rec</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipeline (w/ Tagging)</td>
<td>53.22</td>
<td>54.92</td>
<td>54.05</td>
</tr>
<tr>
<td>Joint w/ Tagging</td>
<td>54.93</td>
<td>54.02</td>
<td>54.47</td>
</tr>
<tr>
<td>Complete Joint</td>
<td>56.06</td>
<td>54.74</td>
<td><strong>55.39</strong></td>
</tr>
</tbody>
</table>

6 Experiments

We use the Automatic Content Extraction (ACE; Doddington et al. (2004)) 2004 English dataset for the experiments, a standard labeled corpus for the three tasks that we are studying. ACE consists of 443 documents from 4 distinct news domains: broadcast, newswire, Arabic treebank, and Chinese treebank. The total number of sentences and tokens are 7,790 and 172,506 respectively. We split the data into train, test, and development sets of sizes 60%, 20%, and 20%. Counts of each type of variable are shown in Table 5.

Our model takes the complete mention string as input, and for these experiments we use the gold mention boundaries. Due to the complexity of inference and the size and density of the model, we restrict the set of labels we explore for both entity tags and relation extraction to the coarse-grained types (7 and 8 respectively).

6.1 Isolated Models

We train the isolated models using the features described in Section 2. Our model for entity tagging achieves an accuracy of 80.2%, which is impressive considering many of the mentions are proper nouns and pronominals with little evidence in the context. Although we do not use sub-types for evaluation, the same model when trained for 42 sub-types achieves 79.7% accuracy. Our relation extraction model achieves an F1 score of 54.05% which is comparable to existing research that uses only predicted entity tags. When using the gold entity tags, the model achieves 61% F1. The coreference model, evaluated by enforcing transitivity as a post-processing step, achieves a macro $B^3$ F1 score of 67.05%, which is competitive with the state of the art when using predicted entity tags.

6.2 Joint Inference Results

We first evaluate joint inference between pairs of tasks, in particular, we separately evaluate the result of joint inference between entity tagging and the two other tasks. The results, when compared to the isolated models, are shown in Tables 7, 8 and 6. Allowing uncertainty in entity tags improves the accuracies of both the tasks, demonstrating the importance of propagating uncertainty along the pipeline. Further, there are significant error reductions for the entity tagging tasks, corroborating the need for flow of information between relation extraction and coreference to the entity tagging model.

When performing inference together on the model defined over all the three tasks, we achieved further improvements for all three tasks, most significantly we achieved an error reduction of 12.4% for the entity tagging task. Even though the model does not directly reflect dependencies between coreference...
<table>
<thead>
<tr>
<th>Model</th>
<th>Pairwise Prec</th>
<th>Rec</th>
<th>F1</th>
<th>(B^3) Prec</th>
<th>Rec</th>
<th>F1</th>
<th>MUC Prec</th>
<th>Rec</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipeline (w/ Tagging)</td>
<td>66.67</td>
<td>51.27</td>
<td>57.96</td>
<td>74.46</td>
<td>60.99</td>
<td>67.05</td>
<td>85.60</td>
<td>61.04</td>
<td>71.26</td>
</tr>
<tr>
<td>Joint w/ Tagging</td>
<td>66.17</td>
<td>52.78</td>
<td>58.72</td>
<td>73.84</td>
<td>62.15</td>
<td>67.49</td>
<td>86.03</td>
<td>61.66</td>
<td>71.84</td>
</tr>
<tr>
<td>Complete Joint</td>
<td>66.33</td>
<td>52.93</td>
<td>58.88</td>
<td>73.91</td>
<td>62.27</td>
<td>67.59</td>
<td>85.91</td>
<td>62.10</td>
<td>72.09</td>
</tr>
</tbody>
</table>

Table 6: Coreference Resolution Results

and relation extraction, these results demonstrate that uncertainty effectively flows across the entity tags such that representing uncertainty in coreference benefits relation extraction, and vice versa.

7 Conclusions and Future Work

This paper introduces a novel, fully-joint modeling of three crucial information extraction tasks, entity tagging, relation extraction, and coreference. The model contains factors that represent the different dependencies that lie between the tasks, resulting in a high tree-width structure containing all the variables of a document. To facilitate efficient inference, we introduce a novel extension to belief propagation that sparsifies variable during inference, effectively eliminating the need to compute majority of the messages. The combination of a joint model, and an accompanying inference technique, allows us to obtain higher accuracies on all the three tasks. These results add substantially to our understanding of the joint inference, providing additional support that the improved representation of multiple tasks in the same model can be beneficial to all the tasks.

Further research might explore incorporation of yet additional tasks onto the current joint inference model. Mention detection, for example, was assumed as an input, however there are direct dependencies between the tasks we model such as coreference and relation extraction that are likely to benefit better detection of boundary detection. As the application of joint inference becomes ubiquitous, there is a strong need for designing efficient inference solutions such as ours that can scale to much larger domains (such as subtypes) and complex tasks (such as transitivity). With a better understanding of joint representations of multiple NLP tasks, and the tools to perform efficient inference for these models, we feel we are on our way to integrate all the subtasks of the information extraction pipeline.

References


Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In Annual Meeting of the Association for Computational Linguistics (ACL), pages...


of information extraction and coreference with application to citation matching. In *Uncertainty in Artificial Intelligence (UAI)*, pages 593–601.


