Unsupervised Relation Extraction with General Domain Knowledge

Oier Lopez de Lacalle[1,2] and **Mirella Lapata**[1]
[1]Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh EH8 9AB
[2]IKERBASQUE, Basque Foundation for Science, Bilbao, Spain
oier.lopezdelacalle@ehu.es, mlap@inf.ed.ac.uk

Abstract

In this paper we present an unsupervised approach to relational information extraction. Our model partitions tuples representing an observed syntactic relationship between two named entities (e.g., "X was born in Y" and "X is from Y") into clusters corresponding to underlying semantic relation types (e.g., *BornIn*, *Located*). Our approach incorporates general domain knowledge which we encode as First Order Logic rules and automatically combine with a topic model developed specifically for the relation extraction task. Evaluation results on the ACE 2007 English Relation Detection and Categorization (RDC) task show that our model outperforms competitive unsupervised approaches by a wide margin and is able to produce clusters shaped by both the data and the rules.

1 Introduction

Information extraction (IE) is becoming increasingly useful as a form of shallow semantic analysis. Learning relational facts from text is one of the core tasks of IE and has applications in a variety of fields including summarization, question answering, and information retrieval. Previous work (Surdeanu and Ciaramita, 2007; Culotta and Sorensen, 2004; Zhou et al., 2007) has traditionally relied on extensive human involvement (e.g., hand-annotated training instances, manual pattern extraction rules, hand-picked seeds). Standard supervised techniques can yield high performance when large amounts of hand-labeled data are available for a fixed inventory of relation types (e.g., *Employment*, *Located*), however, extraction systems do not easily generalize beyond their training domains and often must be re-engineered for each application. Unsupervised approaches offer a promising alternative which could lead to significant resource savings and more portable extraction systems.

It therefore comes as no surprise that latent topic analysis methods have been used for a variety of IE tasks. Yao et al. (2011), for example, propose a series of topic models which perform relation discovery by clustering tuples representing an observed syntactic relationship between two named entities (e.g., "X was born in Y" and "X is from Y"). The clusters correspond to semantic relations whose number or type is not known in advance. Their models depart from standard Latent Dirichlet Allocation (Blei et al., 2003) in that a document consists of relation tuples rather than individual words; moreover, tuples have features each of which is generated independently from a hidden relation (e.g., the words corresponding to the first and second entities, the type and order of the named entities). Since these features are *local*, they cannot capture more global constraints pertaining to the relation extraction task. Such constraints may take the form of restrictions on which tuples should be clustered together or not. For instance, different types of named entities may be indicative of different relations (ORG-LOC entities often express a *Location* relation whereas PER-PER entities express *Business* or *Family* relations) and thus tuples bearing these entities should not be grouped together. Another example are tuples with identical or similar features which intuitively should be clustered together.

In this paper, we propose an unsupervised approach to relation extraction which does not require any relation-specific training data *and* allows to incorporate *global* constraints general expressing domain knowledge. We encode domain knowledge as First Order Logic (FOL) rules and automatically integrate them with a topic model to produce clusters shaped by the data and the constraints at hand. Specifically, we extend the Fold-all (First-Order Logic latent Dirichlet Allocation) framework (Andrzejewski et al., 2011) to the relation extraction task, explain how to incorporate meaningful constraints, and develop a scalable inference technique. In the presence of multiple candidate relation decompositions for a given corpus, domain knowledge can steer the model towards relations which are best aligned with user and task modeling goals. We also argue that a general mechanism for encoding additional modeling assumptions and side information can lessen the need for "custom" relation extraction model variants. Experimental results on the ACE-2007 Relation Detection and Categorization (RDC) dataset show that our model outperforms competitive unsupervised approaches by a wide margin and is able to uncover meaningful relations with only two general rule types.

Our contributions in this work are three-fold: a new model that modifies the Fold-all framework and extends it to the relation extraction task; a new formalization of the logic rules applicable to topic models defined over a rich set of features; and a proposal for mining the logic rules *automatically* from a corpus contrary to Andrzejewski et al. (2011) who employ manually crafted seeds.

2 Related Work

A variety of learning paradigms have been applied to relation extraction. As mentioned earlier, supervised methods have been shown to perform well in this task. The reliance on manual annotation, which is expensive to produce and thus limited in quantity, has provided the impetus for semi-supervised and purely unsupervised approaches. Semi-supervised methods use a small number of seed instances or patterns (per relation) to launch an iterative training process (Riloff and Jones, 1999; Agichtein and Gravano, 2000; Bunescu and Mooney, 2007; Pantel and Pennacchiotti, 2006). The seeds are used to extract a new set of patterns from a large corpus, which are then used to extract more instances, and so on. Unsupervised relation extraction methods are not limited to a predefined set of target relations, but discover all types of relations found in the text. The relations represent clusters over strings of words (Banko et al., 2007; Hasegawa et al., 2004), syntactic patterns between entities (Yao et al., 2011; Shinyama and Sekine, 2006), or logical expressions (Poon and Domingos, 2009). Another learning paradigm is distant supervision which does not require labeled data but instead access to a relational database such as Freebase (Mintz et al., 2009). The idea is to take entities that appear in some relation in the database, find the sentences that express the relation in an unlabeled corpus, and use them to train a relation classifier.

Our own work adds an additional approach into the mix. We use a topic model to infer an arbitrary number of relations between named entities. Although we do not have access to relation-specific information (either as a relational database or manually annotated data), we impose task-specific constraints which inject domain knowledge into the learning algorithm. We thus alleviate known problems with the interpretability of the clusters obtained from topic models and are able to guide our model towards reasonable relations. Andrzejewski et al. (2011) show how to integrate First-Order Logic with vanilla LDA. We extend their formulation to relation tuples rather than individual words. Our model generates a corpus of entity tuples which are in turn represented by features and uses automatically acquired FOL rules. The idea of integrating topic modeling with FOL builds on research in probabilistic logic modeling such as Markov Logic Networks (Richardson and Domingos, 2006). Schoenmackers et al. (2010) learn Horn clauses from web-scale text with aim of finding answers to a user's query. Our work is complementary to theirs. We could make use of their rules to discover more accurate relations.

The general goal of assisting the learner in recovering the "correct" clustering by supplying additional domain knowledge is not new. Gondek and Hofmann (2004) supply a known clustering they do not want the learner to return, whereas Wagstaff et al. (2001) use pairwise labels for items indicating whether they belong in the same cluster. These methods combine domain knowledge with statistical learning in order to improve performance with respect to the true target clustering. Although, the target labels are not available in our case, we are able to show that the inclusion of domain knowledge yields clustering improvements.

3 Learning Setting

Our relation extraction task broadly adheres to the ACE specification guidelines. Our aim is to detect and characterize the semantic relations between two named entities. The input to our model is a corpus of documents, where each document is a bag of relation tuples which can be obtained from the output of any dependency parser. Each tuple represents a syntactic relationship between two named entity (NE) mentions, and consists of three components: the dependency path between the two mentions, the source NE, and the target NE. A dependency path is the concatenation of dependency edges and nodes along a path in the dependency tree. For example, the sentence "***George Bush** traveled to **France** on Thursday for a summit.*" would yield the tuple [SOURCE:*George_Bush*(PER), PATH:*→nsubj→traveled→prep→to→pobj→*, DES:*France*(LOC)]. The tuple here expresses the relation *Located*, however our model does not observe any relation labels during training. The model assigns tuples to clusters, corresponding to an underlying relation type. Each tuple instance can be then labeled with an identifier corresponding to the cluster (aka relation) it has been assigned to.

4 Modeling Framework

Our model builds on the work of Yao et al. (2011) who develop a series of generative probabilistic models for relation extraction. Specifically, we extend their relational LDA model by interfacing it with FOL-rules. In the following, we first describe their approach in more detail and then present our extensions and modifications.

4.1 Relational LDA

Relational LDA is an extension to LDA with a similar generative story. LDA models each document as a mixture of topics, which are in turn characterized as distributions over words. In relational LDA, each document is a mixture of relations over tuples representing syntactic relations between two named entities. The relation tuples are in turn generated a by set of features drawn independently from the underlying relation distribution.

More technically, a multinomial distribution over relations θ_{d_i} is drawn from a Dirichlet prior ($\theta \sim \text{Dir}(\alpha)$) at the document level. Relation tuples are generated from a multinomial distribution θ_{d_i} ($z_i|\theta_{d_i} \sim \text{Mult}(\theta_{d_i})$) and are represented with k features. Each feature is drawn (independently) from a multinomial distribution selected by the relation assigned to tuple i ($f_{ik}|z_i, \phi_{z_i} \sim \text{Mult}(\phi_{z_i})$). Relations are drawn from a Dirichlet prior ($\phi \sim \text{Dir}(\beta)$). In other words, each tuple in a document is assigned a hidden relation ($z = z_1...z_N$); each relation is represented by a multinomial distribution over features ϕ_r (Dirichlet prior β). ϕ_r is a vector with F dimensions each corresponding to a feature. Finally, documents ($j = 1...D$) are associated with a multinomial distribution θ_j over relations (Dirichlet prior α). θ_j is a vector with R dimensions, one for each relation.

Figure 1 represents relational LDA model as a an undirected graphical model or factor graph (Bishop, 2006), ignoring for the moment the factor which connects the $\boldsymbol{d}$, $\boldsymbol{z}$, $f_{1...k}$ and $\boldsymbol{o}$ variables. Directed graphical models can be converted into undirected ones by adding edges between co-parents (Koller and Friedman, 2009). Each clique in the graph defines a potential function which replaces the conditional probabilities in the directed graph. Each maximal clique is associated with a special factor node (the black squares) and clique members are connected to that factor. The probability of any specific configuration is calculated by multiplying the potential functions and normalizing them. We adopt the factor graph representation as is it convenient for introducing logic rules into the model. The joint probability of the model given the priors and the documents ($P(\boldsymbol{p}, \boldsymbol{z}, \phi, \theta|\alpha, \beta, \boldsymbol{d})$) is equivalent to:

$$\prod_{r}^{R} p(\phi_r|\beta) \prod_{j}^{D} p(\theta_j|\alpha) \prod_{i}^{N} \theta_{d_i}(z_i) \prod_{k \in p_i} \phi_{z_i}(f_k) \quad (1)$$

where $\theta_{d_i}(z_i)$ is the z_i-th element in the vector θ_{d_i} and $\phi_{z_i}(f_k)$ is f_k-th feature in the ϕ_{z_i} vector. Variable p_i is the i-th tuple containing k features. The parameters of the latent variables (e.g., ϕ, θ) are typically estimated using an approximate inference algorithm such as Gibbs Sampling (Griffiths and Steyvers, 2004).

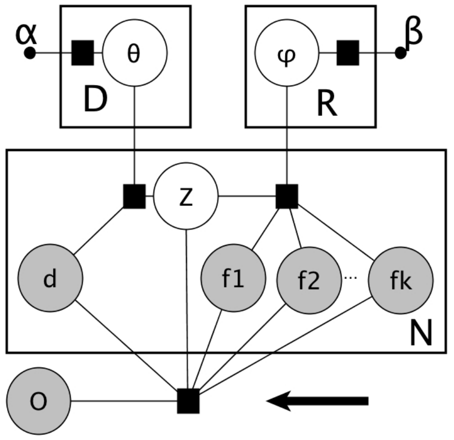

Figure 1: Relational LDA as a factor graph. Filled circles represent observed variables, empty circles are associated with latent variables or model hyperparameters, and plates indicate repeating structures. The black squares are the factor nodes and are associated with the potential functions corresponding to conditional independence among the variables. The model observes D documents ($\boldsymbol{d}$) consisting of N tuples ($\boldsymbol{p}$), each represented by a set of features f_1,$f_2 \ldots f_k$. z represents the relation type assignment to a tuple, θ is the relation type proportion for a given document, and ϕ the relation type distribution over the features. The logic factor (indicated with the arrow) connects the KB with the relational LDA model. Variable $\boldsymbol{o}$ is an observed variable which contains the side information expressed in FOL.

As shown in Figure 1, the observed variables are represented by filled circles. In our case, our model sees the corpus ($\boldsymbol{p}$, $\boldsymbol{d}$), where $\boldsymbol{d}$ is the variable representing the document and the tuples ($\boldsymbol{p}$) are represented by a set of features f_1,$f_2 \ldots f_k$ in the graph. Empty circles are associated with latent variables to be estimated: z represents the relation type assignment to the tuple, θ is the relation type proportion for the given document, and ϕ is the relation type distribution over the features.

The features representing the tuples tap onto semantic information expressed by different surface forms and are an important part of the model. We use a subset of the features proposed in Yao et al. (2011) which we briefly describe below:

SOURCE This feature corresponds to the first entity mention of the tuple. In the sentence ***George Bush*** *traveled to* ***France*** *on Thursday for a summit.*, the value of this feature would be *George Bush*.

Value	Predicate	Description
$z_i = r$	$\mathtt{Z}(i, r)$	Latent relation type
$f_k = v$	$\mathtt{F}(k, v)$	feature of relation tuple
$p_i = i$	$\mathtt{P}(i, f_k)$	tuple i contains feature f_k
$d_i = j$	$\mathtt{D}(i, j)$	observed document

Table 1: Logical variables for Relational LDA. The variable i ranges over tuples in the corpus ($i = [1 \ldots N]$), and k over features in the corpus ($k = [1 \ldots F]$).

DEST The feature corresponds to the second entity mention and its value would be *France* in the previous example.

NEPAIR The feature indicates the type and order of two entity mentions in the tuple. This would be PER-ORG in our example.

PATH This feature refers to the dependency path between two entity mentions. In our sentence, the value of the feature would be PATH*:→nsubj→traveled→prep→to→pobj→*.

TRIGGER Finally, trigger features are content words occurring in the dependency path. The path PATH*:→nsubj→traveled→prep→to→pobj→* contains only one trigger word, namely *traveled*. The intuition behind this feature is that paths sharing the same set of trigger words should be grouped in the same cluster.

4.2 First Order Logic and Relational LDA

We next couple relational LDA with global constraints, which we express using FOL rules. We begin by representing relational LDA as a Markov Logic Network (Richardson and Domingos, 2006). We define a logical predicate for each model variable. For example, assigned relation variable ($\mathtt{Z}(i, r)$) is true if $z_i = r$ and false otherwise. Table 1 shows the mapping of model variables onto logical predicates. Logical rules are encoded in the form of a weighted FOL knowledge base (KB) which is then converted into Conjunctive Normal form:

$$KB = \{(\lambda_1, \psi_1), ..., (\lambda_L, \psi_L)\} \quad (2)$$

The KB consists of L pairs, where each ψ_l represents a FOL rule and $\lambda_l \geq 0$ its weight. Rules are soft preferences rather than hard constraints; the weights represent the importance of ψ_l and are set manually by the domain expert. The KB is tied to the probabilistic model via its *groundings* in the corpus. For each FOL rule ψ_l, let $G(\psi_l)$ be the set of groundings, each mapping the free variables in ψ_l to a specific value. For example, in the rule $\forall i, j, p : \texttt{F}(i, \mathit{Obama}) \wedge \texttt{F}(j, \mathit{WhiteHouse}) \wedge \texttt{P}(p, i) \wedge \texttt{P}(p, j) \Rightarrow \texttt{Z}(p, r)$[1], G consists of all the rules where the free variables i, j and p are instantiated. At grounding time, we parse the corpus searching for the tuples that satisfy the logic rules and store the indices of the tuples that ground the rule. The stored indices are used to set ψ_l to a specific value. For the (*Obama, White House*) example above, G consists of F propositional rules for each observed feature, where $i \in [1 \dots F]$. For each grounding ($g \in G(\psi_l)$) we define an indicator function:

$$\mathbb{1}_g(\boldsymbol{z}, \boldsymbol{p}, \boldsymbol{d}, \boldsymbol{o}) = \begin{cases} 1, & \text{if } g \text{ is true under} \\ & \boldsymbol{z} \text{ and } \boldsymbol{p}, \boldsymbol{d}, \boldsymbol{o} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where $\boldsymbol{z}$ are relation assignments to tuples, $\boldsymbol{p}$ is the set of features in tuples, $\boldsymbol{d}$ are documents, and $\boldsymbol{o}$ the side information encoded in FOL. Contrary to Andrzejewski et al. (2011), we need to ground the rules while taking into account if the feature specified in the rule is expressed by any tuple or the specific given tuple, since we are assigning relations to tuples, and not directly to words.

Next, we define a Markov Random Field (MRF) which combines relational LDA with the FOL knowledge base. The MRF is defined over latent relation tuple assignments $\boldsymbol{z}$, relation feature multinomials ϕ, and relation document multinomials θ (the feature set, document, and external information $\boldsymbol{o}$ are observed). Under this model the conditional probability $P(\boldsymbol{z}, \phi, \theta | \alpha, \beta, \boldsymbol{p}, \boldsymbol{d}, \boldsymbol{o}, KB)$ is proportional to:

$$\exp\left(\sum_l^L \sum_{g \in G(\psi_l)} \lambda_l \mathbb{1}_g(\boldsymbol{z}, \boldsymbol{p}, \boldsymbol{d}, \boldsymbol{o})\right) \times \\ \prod_r^R p(\phi_r|\beta) \prod_j^D p(\theta_j|\alpha) \prod_i^N \theta_{d_i}(z_i) \prod_{k \in p_i} \phi_{z_i}(f_k) \tag{4}$$

The first term in Equation (4) corresponds to the logic factor in Figure 1 that groups variables $\boldsymbol{d}$, $\boldsymbol{z}$, $f_1, f_2, \dots f_k$ and $\boldsymbol{o}$. The remaining terms in Equation (4) refer to relational LDA. The goal of the model is to estimate the most likely θ and ϕ for the given observed state. As $\boldsymbol{z}$ can not be marginalized out, we proceed with MAP estimation of ($\boldsymbol{z}$, ϕ, θ), maximizing the log of the probability as in Andrzejewski et al. (2011):

$$\operatorname*{arg\,max}_{\boldsymbol{z},\phi,\theta} \sum_l^L \sum_{g \in G(\psi_l)} \lambda_l \mathbb{1}_g(\boldsymbol{z}, \boldsymbol{p}, \boldsymbol{d}, \boldsymbol{o}) + \\ \sum_r^R \log p(\phi_r|\beta) + \\ \sum_i^N \log \theta_{d_i}(z_i) \prod_{k \in p_i} \phi_{z_i}(f_k) \tag{5}$$

Once the parameters of the model are estimated (see Section 4.3 for details), we use the ϕ probability distribution to assign a relation to a new test tuple. We select the relation that maximizes the probability $\arg\max_r \prod_i^k P(f_i|\phi_r)$ where $f_1 \dots f_k$ are features representing the tuple and r the relation index.

4.3 Inference

Exact inference is intractable for both relational LDA and MLN models. In order to infer the most likely multinomial parameters ϕ and θ, we applied the Alternating Optimization with Mirror Descent algorithm introduced in Andrzejewski et al. (2011). The algorithm alternates between optimizing the multinomial parameters (ϕ, θ), whilst holding the relation assignments ($\boldsymbol{z}$) fixed, and vice-versa. At each iteration, the algorithm first finds the optimal (ϕ, θ) for a fixed $\boldsymbol{z}$ as the MAP estimate of the Dirichlet posterior:

$$\phi_r(f) \propto n_{rf} + \beta - 1 \tag{6}$$

$$\theta_j(r) \propto n_{jr} + \alpha - 1 \tag{7}$$

where n_{rf} is the number of times feature f is assigned to relation r in relation assignments $\boldsymbol{z}$, and n_{jr} is the number of times relation r is assigned to document j. Next, $\boldsymbol{z}$ is optimized while keeping ϕ and θ fixed. This step is divided into two parts. The algorithm first deals with all z_i which appear only in trivial groundings, i.e., groundings whose indicator functions $\mathbb{1}_g$ are not affected by the latent relation assignment $\boldsymbol{z}$. As z_i only appears in the last term of

[1]This rule translates as "every tuple containing *Obama* and *White House* as features should be in relation cluster r".

Equation (5), the algorithm needs only optimize the following term:

$$z_i = \arg\max_{r=1\ldots R} \theta_{d_i}(r) \prod_{k \in p_i} \phi_{z_i}(f_k) \tag{8}$$

The second part deals with the remaining z_i that appear in non-trivial groundings in the first term of Equation (5). We follow Andrzejewski et al. (2011) in relaxing (5) into a continuous optimization problem and refer the reader to their paper for a more in depth treatment. Suffice it to say that once the binary variables $z_{ir} \in \{0, 1\}$ are relaxed to continuous values $z_{ir} \in [0, 1]$, it is possible to introduce the relational LDA term in the equation and compute the gradient using the Entropic Mirror Descent Algorithm (Beck and Teboulle, 2003):

$$\begin{gathered}\arg\max_{\boldsymbol{z} \in [0,1]^{|KB|}} \sum_{l}^{L} \sum_{g \in G(\psi_l)} \lambda_l \mathbb{1}_g(\boldsymbol{z}) + \\ \sum_{i,r} z_{ir} \log \theta_{d_i}(r) \prod_{k \in p_i} \phi_{z_i}(f_k) \\ \text{s.t } z_{ir} \geq 0\ ,\ \sum_{i,r} z_{ir} = 1\end{gathered} \tag{9}$$

In every iteration the approximation algorithm randomly samples a term from the objective function (Equation (9)). The sampled term can be a particular ground rule g or the relational LDA term $(\sum_r z_{ir} \log \theta_{d_i}(r) \prod_{k \in p_i} \phi_{z_i}(f_k))$ for some uniformly sampled index i. The sampling of the terms is weighted according to the rule weight (λ_l) and the grounded value $(G(\psi_l))$ in the case of logic rules, and the size of corpus in tuples $(|\boldsymbol{z}_{KB}|)$ for relational LDA. Once we choose term f and take the gradient, we can apply the Entropic Mirror Descent update:

$$z_{ir} \leftarrow \frac{z_{ir} \exp(\eta \nabla_{z_{ir}} f)}{\sum_{r'} z_{ir'} \exp(\eta \nabla_{z_{ir'}} f)} \tag{10}$$

Finally, z_i is recovered by rounding to $\arg\max_r z_{ir}$. The main advantage of this approach is that it requires only a means to sample groundings g for each rule ψ_l, and can avoid fully grounding the FOL rules.

4.4 Logic Rules

Our model assigns relations to tuples rather than topics to words. Since our tuples are described in terms of features our logic rules must reflect this too. For our experiments we defined two very general types of rules described below.

Must-link Tuple The motivation behind this rule is that tuples which share features probably express the same underlying relation. The rule must specify which feature has to be shared for the tuples to be clustered together. For example, the rule below states that tuples containing the dependency path PATH:*→appos→president→prep→of→pobj→* should go in the same cluster:

$$\begin{gathered}\forall i,j,k : \mathtt{F}(i, \text{PATH:}\textit{is\_the\_president\_of}) \land \mathtt{P}(j, f_i) \\ \land \mathtt{P}(k, f_i) \Rightarrow \neg \mathtt{Z}(j,t) \lor \mathtt{Z}(k,r)\end{gathered}$$

Cannot-link Tuple We also define rules prohibiting tuples to be clustered together because they do not share any features. For example, tuples with ORG-LOC entities, probably express a *Location* relation and should not be clustered together with PER-PER tuples, which in all likelihood express a different relationship (e.g., *Family*). The rule below expresses this constraint:

$$\begin{gathered}\forall i,j,k,l : \mathtt{F}(i, \text{NEPAIR:}\textit{PER-PER}) \\ \land \mathtt{F}(j, \text{NEPAIR:}\textit{ORG-LOC}) \\ \land \mathtt{P}(k, f_i) \land \mathtt{P}(l, f_j) \Rightarrow \neg \mathtt{Z}(k,r) \lor \neg \mathtt{Z}(l,r)\end{gathered}$$

The specification of the first order logic rules is an integral part of the model. The rules express knowledge about the task at hand, the domain involved, and the way the relation extraction problem is modeled (i.e., tuples expressed as features). So far, we have abstractly formulated the rules without explaining how they are specifically instantiated in our model. We could write them down by hand after inspecting some data or through consultation with a domain expert. Instead, we obtain logic rules *automatically* from a corpus following the procedure described in Section 5.

5 Experimental Setup

Data We trained our model on the New York Times (years 2000–2007) corpus created by Yao et al. (2011). The corpus contains approximately 2M entity tuples. The latter were extracted from 428K documents. After post-processing (tokenization, sentence-splitting, and part-of-speech tagging),

Must-link Tuple
$\mathtt{F}(i, \text{NEPAIR:}\textit{PER-PER}, \text{TRIGGER:}\textit{wife}) \wedge \mathtt{P}(j, f_i) \wedge \mathtt{P}(k, f_i) \Rightarrow \neg\mathtt{Z}(j, t) \vee \mathtt{Z}(k, r)$
$\mathtt{F}(i, \text{NEPAIR:}\textit{PER-LOC}, \text{TRIGGER:}\textit{die}) \wedge \mathtt{P}(j, f_i) \wedge \mathtt{P}(k, f_i) \Rightarrow \neg\mathtt{Z}(j, t) \vee \mathtt{Z}(k, r)$
$\mathtt{F}(i, \text{PATH:}\leftarrow\textit{nsubj}\leftarrow\textit{die}\rightarrow\textit{prep}\rightarrow\textit{in}\rightarrow\textit{pobj}\rightarrow) \wedge \mathtt{P}(j, f_i) \wedge \mathtt{P}(k, f_i) \Rightarrow \neg\mathtt{Z}(j, t) \vee \mathtt{Z}(k, r)$
$\mathtt{F}(i, \text{SOURCE:}\textit{Kobe}, \text{DEST:}\textit{Lakers}) \wedge \mathtt{P}(j, f_i) \wedge \mathtt{P}(k, f_i) \Rightarrow \neg\mathtt{Z}(j, t) \vee \mathtt{Z}(k, r)$
Cannot-link Tuple
$\mathtt{F}(i, \text{NEPAIR:}\textit{ORG-LOC}) \wedge \mathtt{F}(j, \text{NEPAIR:}\textit{PER-PER}) \wedge \mathtt{P}(k, f_i) \wedge \mathtt{P}(l, f_j) \Rightarrow \neg\mathtt{Z}(k, r) \vee \neg\mathtt{Z}(l, r)$
$\mathtt{F}(i, \text{NEPAIR:}\textit{LOC-LOC}) \wedge \mathtt{F}(j, \text{TRIGGER:}\textit{president}) \wedge \mathtt{P}(k, f_i) \wedge \mathtt{P}(l, f_j) \Rightarrow \neg\mathtt{Z}(k, r) \vee \neg\mathtt{Z}(l, r)$
$\mathtt{F}(i, \text{NEPAIR:}\textit{PER-LOC}) \wedge \mathtt{F}(j, \text{TRIGER:}\textit{member}) \wedge \mathtt{P}(k, f_i) \wedge \mathtt{P}(l, f_j) \Rightarrow \neg\mathtt{Z}(k, r) \vee \neg\mathtt{Z}(l, r)$
$\mathtt{F}(i, \text{NEPAIR:}\textit{PER-PER}) \wedge \mathtt{F}(j, \text{TRIGER:}\textit{sell}) \wedge \mathtt{P}(k, f_i) \wedge \mathtt{P}(l, f_j) \Rightarrow \neg\mathtt{Z}(k, r) \vee \neg\mathtt{Z}(l, r)$

Table 2: Examples of automatically extracted Must-link and Cannot-link tuple rules.

named entities were automatically recognized and labeled with PER, ORG, LOC, and MISC (Finkel et al., 2005). Dependency paths for each pair of named entity mentions were extracted from the output of the MaltParser (Nivre et al., 2004). In our experiments, we discarded tuples with paths longer than 10 edges (Lin and Pantel, 2001). We evaluated our model on the test partition of the ACE 2007 (English) RDC dataset which is labeled with gold standard entity mentions and their relations. There are six general relation types and 18 subtypes. We used 25% of the ACE training partition as a development set for parameter tuning.

Logic Rule Extraction We automatically extracted logic rules from the New York Times (NYT) corpus as follows. The intuition behind Must-link rules is that tuples with common features should cluster together. Although we do not know which features would yield the best rules, we naively assume that good features are frequently co-occurring features. Using the log-likelihood ratio (Dunning, 1993), we first discarded low confidence feature co-occurrences ($p < 0.05$). Two features co-occur if they are both found within the same sentence. We then sorted the remaining co-occurrences by their frequency and retained the N-best ones. We only considered unigram and bigram features since higher-order ones tend to be sparse. An example of a bigram feature would be (PATH:←nsubj←grow→prep→in→pobj→, DEST:Chicago).

The main intuition behind Cannot-link rules is that tuples without any common features should not cluster together. So, if two features never co-occur, they probably express different relations. For every unigram and bigram feature in the respective N-best list, we find the features it does not co-occur with in the NYT corpus. For example, NEPAIR:PER–LOC does not co-occur with DEST:*Yankees* and the bigram DEST:*United Nations*, NEPAIR:PER–ORG does not co-occur with SOURCE:Mr. Bush, NEPAIR:PER–LOC. Cannot-link rules are then based on such non-co-occurring feature pairs.

We optimized N empirically on the development set. We experimented with values ranging from 20 to 500. We obtained 20 Must-link rules for coarse-grained relations and 400 rules for their subtypes. We extracted 1,814 Cannot-link rules for general relations ($N = 50$) and 34,522 rules for subtypes ($N = 400$). The number of features involved in the Must-link rules was 25 for coarse-grained relations and 422 for fine-grained relations. For Cannot-link rules, 62 features were involved in coarse-grained relations and 422 in fine-grained relations.

Examples of the rules we extracted are shown in Table 2. The first rule in the upper half of the table states that tuples must cluster together if their source and target entities are PER and contain the trigger word *wife* in their dependency path. The second rule is similar, the source entity here is PER, the target LOC and the trigger word is *die*. According to the third rule, tuples featuring the path PATH:←*nsubj*←*die*→*prep*→*in*→*pobj*→ should be in the same cluster. The fourth rule forces tuples whose source entity is *Kobe* and target entity is *Lakers* to cluster together. The second half of the table illustrates Cannot-link tuple rules. The first rule prevents tuples with ORG-LOC entities to cluster together with PER-PER tuples. The second rule states that we cannot link LOC-LOC tuples with those whose trigger word is *president*, and so on.

Parameter Tuning Our framework has several parameters that must be adjusted for an optimal clustering solution. These include the hyperparameters α and β as well as the number of clusters. In addition, we have to assign a weight to each FOL rule grounding. An exhaustive search on the hyperparameters and rule weights is not possible. We therefore followed a step-wise approximation procedure. First, we find the best α and β values, whilst varying the number of clusters. Once we have the best hyperparameters for each clustering, we set the weights for the FOL rules. We varied the number of relations from 5 to 50. We experimented with α values in the range of $[0.05 - 0.5]$ and β values in the range of $[0.05 - 0.5]$. These values were optimized separately for coarse- and fine-grained relations. Table 3 shows the optimal number of clusters for different model variants and relation types.

The FOL weights can also make a difference in the final output; the bigger the weight the more times the rule will be sampled in the Mirror Descent algorithm. We experimented with two weighting schemes: (a) we gave a weight of 1 or 0.5 to each rule grounding and (b) we scaled the weights so as to make their contribution comparable to relational LDA. We obtained best results on the development set with the former scheme.

Baselines We compared our FOL relational LDA model against standard LDA (Blei et al., 2003) and relational LDA without the FOL component. In the case of standard LDA, we estimated topics (relations) over words, and used the context of the entity mentions pairs as a bag of words feature to select the most likely cluster at test time. Parameters for LDA and relational LDA were optimized following the same parameter tuning procedure described above.

We also compared our model against the unsupervised method introduced in Hasegawa et al. (2004). Their key idea is to cluster pairs of co-occurring named entities according to the similarity of their surrounding contexts. Following their approach, we measured context similarity using the vector space model and the cosine metric and grouped NE pairs into clusters using a complete linkage hierarchical clustering algorithm. We adopted the same parameter values as detailed in their paper (e.g., cosine similarity threshold, length of context vectors). At test time, instances were assigned to the relation cluster most similar to them (according to the cosine measure).

Evaluation We evaluated the clusters obtained by our model and the comparison systems using the Fscore measure introduced in the SemEval 2007 task (Agirre and Soroa, 2007); it is the harmonic mean of precision and recall defined as the number of correct members of a cluster divided by the number of items in the cluster and the number of items in the gold-standard class, respectively.

6 Results

Our results are summarized in Table 3 which reports Fscore for (Hasegawa et al., 2004), LDA, relational LDA (RelLDA), and our model with the FOL component. To assess the impact of the rules on the clustering, we conducted several rule ablation studies. We thus present results with a model that includes both Must-link and Cannot-link tuple rules (CLT+MLT), and models that include either Must-link (MLT) or Cannot-link (CLT) rules but not both. We show the performance of these models with the entire feature set (see (ALL) in the table) and with a subset consisting solely of NE pair related features (see (NEPAIR) in the table). We report results against coarse- and fine-grained relations (6 and 18 relation types in ACE, respectively). The table shows the optimal number of relation clusters (in parentheses) per model and relation type.

We also wanted to examine the quality of the logic rules. Recall that we learn these heuristically from the NYT corpus. We thus trained an additional variant of our model with rules extracted from the ACE training set (75%) which contains relation annotations. The extraction procedure was similar to the unsupervised case, save that the relation types were known and thus informative features could be mined more reliably. For Must-link rules, we extracted unigram and bigram feature frequencies for each relation type and applied TF-IDF weighting in order to discover the most discriminative ones. We created logic rules for the 10 best feature combinations in each relation type. Regarding Cannot-link rules, we enumerated the features (unigrams and bigrams)

Model	Subtype		Type	
HASEGAWA	26.1	(12)	34.7	(12)
LDA	23.4	(10)	29.0	(5)
RelLDA	30.4	(40)	38.6	(5)
U-MLT (ALL)	36.6	(10)	48.0	(5)
U-CLT (ALL)	30.5	(5)	39.3	(5)
U-CLT+MLT (ALL)	29.8	(5)	42.0	(5)
U-MLT (NEPAIR)	36.5	(10)	47.2	(5)
U-CLT (NEPAIR)	28.8	(50)	40.5	(5)
U-CLT+MLT (NEPAIR)	30.9	(10)	41.5	(5)
S-MLT (ALL)	37.0	(10)	47.0	(5)
S-CLT (ALL)	31.4	(50)	40.9	(5)
S-CLT+MLT (ALL)	32.3	(10)	42.5	(5)
S-MLT (NEPAIR)	37.0	(10)	47.6	(10)
S-CLT (NEPAIR)	31.4	(10)	40.1	(5)
S-CLT+MLT (NEPAIR)	37.1	(10)	46.0	(5)

Table 3: Model performance on the ACE 2007 test set using Fscore. Results are shown for six main relation types and their subtypes (18 in total). (ALL) models contain rules extracted from the entire feature set. For (NEPAIR) models, rules were extracted from NEPAIR-related features only. Prefix U- denotes models that use unsupervised rules; prefix S- highlights models using supervised rules. The optimal number of relations per model is shown in parentheses.

that did not co-occur in any relation type and applied TF-IDF weighting. Again, we created rules for the 10 most discriminative features. We defined rules over the entire feature set (466 Must-link and 26,074 Cannot-link rules) and a subset containing only NE pairs. In Table 3, prefixes S- and U- indicate model variants with supervised and unsupervised rules, respectively.

Our results show that standard LDA is not suitable for relation extraction. The obtained clusters are not informative enough to induce semantic relations, whereas RelLDA yields substantially better Fscores. This is not entirely surprising, given that RelLDA is a relation extraction specific model. Hasegawa et al.'s (2004) model lies somewhere in the middle between LDA and RelLDA. The combination of RelLDA with automatically extracted FOL rules improves over RelLDA across the board (see the U- models in Table 3). MLT rules deliver the largest improvement for both coarse and fine-grained relation types. In general, CLT models perform worse as well as models using both types of rules (MLT+CLT). The inferior performance of the rule combination may be due to the fact that MLT and CLT rules contain conflicting information and to a certain extent cancel each other out. The use of many rules might also negatively impact inference, i.e., discriminative rules are sampled less and cannot influence the model towards a better solution. Restricting the number of features and rules to named entity pairs only incurs a negligible drop in performance. This is good news for scaling purposes, since a small number of rules can greatly speed-up inference. Interestingly, model variants which use supervised FOL rules (see the prefix S- in Table 3) perform on par with unsupervised models. Again, MLT rules perform best in the supervised case, whereas CLT rules marginally improve over RelLDA.

We assessed whether differences in performance are statistically significant ($p < 0.05$) using bootstrap resampling (Noreen, 1989). All models across all relation types are significantly better than LDA and Hasegawa et al. (2004). FOL-based models perform significantly better than RelLDA, with the exception of all CLT models and U-CLT+MLT (ALL). MLT models are significantly better than any other rule-based model, except those that only use NEPAIR features. We also measured whether different models agree on their topic assignments using Cohen's Kappa.[2] RelLDA agrees least with MLT models and most with CLT models (i.e., $\kappa = 0.50$ for U-MLT (ALL) and $\kappa = 0.65$ for U-CLT (ALL)). This suggests that the CLT rules do not affect the output of RelLDA as much as MLT ones. Examples of relation clusters discovered by the U-MLT (ALL) model are shown in Table 4.

A last note on parameter selection. Our experiments explored the parameter space extensively in order to examine any interactions between the induced relations and the logic rules. For most model variants inferring subtype relations, the preferred number of clusters is 10. For coarse-grained relations, the optimal number of clusters is five. Overall, we found that the quality of the output is highly correlated with the quality of the logic rules and that a few *good* rules are more important than the optimal number of clusters. We consider these findings robust enough to apply across domains and datasets.

[2]For all comparison models the number of relation clusters was set to 10.

	SOURCE	PATH	DEST
Employment	Republican	president of	Senate
	Senate	director of	Yankees
	House	professor at	Republican
	Bush	chairman of	Congress
	Democrat	spokesman for	House
	Mr. Bush	executive of	Mets
	Democrats	director at	U. of California
	Republican	analyst at	United Nations

	SOURCE	PATH	DEST
Sports	Yankees	defeat	World Series
	Mets	win	Olympic
	United States	beat	World Cup
	Giants	play	Yankees
	Jets	win	Super Bowl
	Nets	lose	Olympics
	Knicks	sign	Mets
	Rangers	victory over	Giants

Table 4: Clusters discovered by the U-MLT (ALL) model indicating employment- and sports-type relations. For the sake of readability, we do not display the syntactic dependencies between words in a path.

7 Conclusions

In this paper we presented a new model for unsupervised relation extraction which operates over tuples representing a syntactic relationship between two named entities. Our model clusters such tuples into underlying semantic relations (e.g., *Located*, *Family*) by incorporating general domain knowledge which we encode as First Order Logic rules. Specifically, we combine a topic model developed for the relation extraction task with domain relevant rules, and present an algorithm for estimating the parameters of this model. Evaluation results on the ACE 2007 (English) RDC task show that our model outperforms competitive unsupervised approaches by a wide margin and is able to produce clusters shaped by both the data and the rules.

In the future, we would like to explore additional types of rules such as seed rules, which would assign tuples complying with the "seed" information to distinct relations. Aside from devising new rule types, an obvious next step would be to explore different ways of extracting the rule set based on different criteria (e.g., the most general versus most specific rules). Also note that in the current framework rule weights are set manually by the domain expert. An appealing direction would be to learn these automatically e.g., via a procedure that optimizes some clustering objective. Finally, it should be interesting to use some form of distant supervision (Mintz et al., 2009) either as a means of obtaining useful rules or to discard potentially noisy or uninformative rules.

Acknowledgments

We gratefully acknowledge financial support from the Department of Education, Universities and Research of the Basque Government (BFI-2011-442). We also thank Limin Yao and Sebastian Riedel for sharing their corpus with us and the members of the Probabilistic Models reading group at the University of Edinburgh for helpful feedback.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the 5th ACM International Conference on Digital Libraries*, pages 85–94, San Antonio, Texas.

Eneko Agirre and Aitor Soroa. 2007. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 7–12, Prague, Czech Republic.

David Andrzejewski, Xiaojin Zhu, Mark Craven, and Ben Recht. 2011. A framework for incorporating general domain knowledge into latent Dirichlet allocation using first-order logic. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 1171–1177, Barcelona, Spain.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2670–2676, Hyderabad, India.

Amir Beck and Marc Teboulle. 2003. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Razvan Bunescu and Raymond Mooney. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th Annual Meeting* *of the Association of Computational Linguistics*, pages 576–583, Prague, Czech Republic.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics, Main Volume*, pages 423–429, Barcelona, Spain.

Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 363–370, Ann Arbor, Michigan.

David Gondek and Thomas Hofmann. 2004. Non-redundant data clustering. In *IEEE International Conference on Data Mining*, pages 75–82. IEEE Computer Society.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *PNAS*, 101(1):5228–5235.

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 415–422, Barcelona, Spain.

D. Koller and N. Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Dekang Lin and Patrick Pantel. 2001. DIRT – discovery of inference rules from text. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 323–328, San Francisco, California.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of the 8th Conference on Computational Natural Language Learning*, pages 49–56, Boston, Massachusetts.

Eric W. Noreen. 1989. *Computer-intensive Methods for Testing Hypotheses: An Introduction*. John Wiley and Sons Inc.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 113–120, Sydney, Australia.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Suntec, Singapore.

Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine Learning*, 62(1–2):107–136.

Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 474–479, Stockholm, Sweden.

Stefan Schoenmackers, Jesse Davis, Oren Etzioni, and Daniel Weld. 2010. Learning first-order Horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1088–1098, Cambridge, MA, October. Association for Computational Linguistics.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 304–311, New York City, USA.

Mihai Surdeanu and Massimiliano Ciaramita. 2007. Robust information extration with perceptrons. In *Proceedings of the NIST 2007 Automatic Content Extraction Workshop*.

Kiri Wagstaff, Claire Cardie, C Rogers, and S Schrödl. 2001. Constrained k-means clustering with background knowledge. In *International Conference on Machine Learning*, pages 577–584. Morgan Kaufmann.

Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1456–1466, Edinburgh, Scotland, UK.

GuoDong Zhou, Min Zhang, DongHong Ji, and QiaoMing Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 728–736, Prague, Czech Republic.