Structured Relation Discovery using Generative Models

Limin Yao[*] **Aria Haghighi**[+] **Sebastian Riedel**[*] **Andrew McCallum**[*]
[*] Department of Computer Science, University of Massachusetts at Amherst
[+] CSAIL, Massachusetts Institute of Technology
{lmyao,riedel,mccallum}@cs.umass.edu
{aria42}@csail.mit.edu

Abstract

We explore unsupervised approaches to relation extraction between two named entities; for instance, the semantic *bornIn* relation between a person and location entity. Concretely, we propose a series of generative probabilistic models, broadly similar to topic models, each which generates a corpus of observed triples of entity mention pairs and the surface syntactic dependency path between them. The output of each model is a clustering of observed relation tuples and their associated textual expressions to underlying semantic relation types. Our proposed models exploit entity type constraints within a relation as well as features on the dependency path between entity mentions. We examine effectiveness of our approach via multiple evaluations and demonstrate 12% error reduction in precision over a state-of-the-art weakly supervised baseline.

1 Introduction

Many NLP applications would benefit from large knowledge bases of relational information about entities. For instance, knowing that the entity *Steve Balmer* bears the *leaderOf* relation to the entity *Microsoft*, would facilitate question answering (Ravichandran and Hovy, 2002), data mining, and a host of other end-user applications. Due to these many potential applications, relation extraction has gained much attention in information extraction (Kambhatla, 2004; Culotta and Sorensen, 2004; Mintz et al., 2009; Riedel et al., 2010; Yao et al., 2010). We propose a series of generative probabilistic models, broadly similar to standard topic models, which generate a corpus of observed triples of entity mention pairs and the surface syntactic dependency path between them. Our proposed models exploit entity type constraints within a relation as well as features on the dependency path between entity mentions. The output of our approach is a clustering over observed relation paths (e.g. "X was born in Y" and "X is from Y") such that expressions in the same cluster bear the same semantic relation type between entities.

Past work has shown that standard supervised techniques can yield high-performance relation detection when abundant labeled data exists for a fixed inventory of individual relation types (e.g. *leaderOf*) (Kambhatla, 2004; Culotta and Sorensen, 2004; Roth and tau Yih, 2002). However, less explored are *open-domain* approaches where the set of possible relation types are not fixed and little to no labeled is given for each relation type (Banko et al., 2007; Banko and Etzioni, 2008). A more related line of research has explored inducing relation types via clustering. For example, DIRT (Lin and Pantel, 2001) aims to discover different representations of the same semantic relation using distributional similarity of dependency paths. Poon and Domingos (2008) present an Unsupervised semantic parsing (USP) approach to partition dependency trees into meaningful fragments (or "parts" to use their terminology). The combinatorial nature of this dependency partition model makes it difficult for USP to scale to large data sets despite several necessary approximations during learning and infer-

Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 1456–1466,
Edinburgh, Scotland, UK, July 27–31, 2011.

ence. Our work is similar to DIRT and USP in that we induce relation types from observed dependency paths, but our approach is a straightforward and principled generative model which can be efficiently learned. As we show empirically, our approach outperforms these related works when trained with the same amount of data and further gains are observed when trained with more data.

We evaluate our approach using 'intrinsic' clustering evaluation and 'extrinsic' evaluation settings.[1] The former evaluation is performed using subset of induced clusters against Freebase relations, a large manually-built entity and relational database. We also show some clusters which are not included as Freebase relations, as well as some entity clusters found by our approach. The latter evaluation uses the clustering induced by our models as features for relation extraction in distant supervision framework. Empirical results show that we can find coherent clusters. In relation extraction, we can achieve 12% error reduction in precision over a state-of-the-art weakly supervised baseline and we show that using features from our proposed models can find more facts for a relation without significant accuracy loss.

2 Problem and Experimental Setup

The task of relation extraction is mapping surface textual relations to underlying semantic relations. For instance, the textual expression "X was born in Y" indicates a semantic relation *bornIn* between entities "X" and "Y". This relation can be expressed textually in several ways: for instance, "X, a native of Y" or "X grew up in Y". There are several components to a coherent relation type, including a tight small number of textual expressions as well as constraints on the entities involved in the relation. For instance, in the *bornIn* relation "X" must be a person entity and "Y" a location (typically a city or nation). In this work, we present an unsupervised probabilistic generative model for inducing clusters of relation types and recognizing their textual expressions. The set of relation types is not pre-specified but induced from observed unlabeled data. See Table 4 for examples of learned semantic relations.

Our observed data consists of a corpus of documents and each document is represented by a bag of relation tuples. Each tuple represents an observed syntactic relationship between two Named Entities (NE) and consists of three components: the dependency path between two NE mentions, the source argument NE, and the destination argument NE. A dependency path is a concatenation of dependency relations (edges) and words (nodes) along a path in a dependency tree. For instance, the sentence "John Lennnon was born in Liverpool" would yield the relation tuple (*Lennon*, $[\uparrow -nsubjpass, born, \downarrow -in]$, *Liverpool*). This relation tuple reflects a semantic *bornIn* relation between the *John Lennon* and *Liverpool* entities. The dependency path in this example corresponds to the "X was born in Y" textual expression given earlier. Note that for the above example, the *bornIn* relation can only occur between a *person* and a *location*. The relation tuple is the primary observed random variable in our model and we construct our models (see Section 3) so that clusters consist of textual expressions representing the same underlying relation type.

3 Models

We propose three generative models for modeling tuples of entity mention pairs and the syntactic dependency path between them (see Section 2). The first two models, Rel-LDA and Rel-LDA1 are simple extensions of the standard LDA model (Blei et al., 2003). At the document level, our model is identical to standard LDA; a multinomial distribution is drawn over a fixed number of relation types R. Changes lie in the observations. In standard LDA, the atomic observation is a word drawn from a latent topic distribution determined by a latent topic indicator variable for that word position. In our approach, a document consists of an exchangeable set of relation tuples. Each relation tuple is drawn from a relation type 'topic' distribution selected by a latent relation type indicator variable. Relation tuples are generated using a collection of independent features drawn from the underlying relation type distribution. These changes to standard LDA are intended to have the effect that instead of representing semantically related words, the 'topic' latent variable represents a relation type.

Our third model exploits entity type constraints within a relation and induces clusters of relations

[1] See Section 4 for a fuller discussion of evaluation.

and entities jointly. For each tuple, a set of relation level features and two latent entity type indicators are drawn independently from the relation type distribution; a collection of entity mention features for each argument is drawn independently from the entity type distribution selected by the entity type indicator.

Path	X, made by Y
Source	Gamma Knife
Dest	Elekta
Trigger	make
Lex	, made by the Swedish medical technology firm
POS	, VBN IN DT JJ JJ NN NN
NER pair	MISC-ORG
Sync pair	partmod-pobj

Table 1: The features of tuple '(Gamma Knife, made by, Elekta)' in sentence "Gamma Knife, made by the Swedish medical technology firm Elekta, focuses low-dosage gamma radiation ..."

$\|R\|$	Number of relations
$\|D\|$	Number of documents
r	A relation
doc	A document
p, s, d	Dep path, source and dest args
f	A feature/feature type
T	Entity type of one argument
α	Dirichlet prior for θ_{doc}
β_x	Dirichlet prior for ϕ_{rx}
β	Dirichlet prior for ϕ_t
θ_{doc}	$p(r\|doc)$
ϕ_{rx}	$p(x\|r)$
ϕ_t	$p(f_s\|T)$, $p(f_d\|T)$

Table 2: The notation used in our models

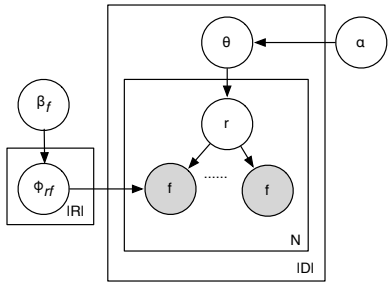

Figure 1: Rel-LDA model. Shaded circles are observations, and unshaded ones are hidden variables. A document consists of N tuples. Each tuple has a set of features. Each feature of a tuple is generated independently from a hidden relation variable r.

3.1 Rel-LDA Model

This model is an extension to the standard LDA model. At the document level, a multinomial distribution over relations θ_{doc} is drawn from a prior Dir(α). To generate a relation tuple, we first draw a relation 'topic' r from Multi(θ). Then we generate each feature f of a tuple independently from a multinomial distribution Multi(ϕ_{rf}) selected by r. In this model, each tuple has three features, i.e. its three components, shown in the first three rows in Table 1. Figure 1 shows the graphical representation of Rel-LDA. Table 2 lists all the notation used in describing our models.

The learning process of the models is an EM process. The procedure is similar to that used by the standard topic model. In the variational E-step (inference), we sample the relation type indicator for each tuple using $p(r|f)$:

$$P(r|f(p,s,d)) \propto p(r)\prod_f p(f|r)$$
$$\propto (\alpha_r + n_{r|d})\prod_f \frac{\beta_f + n_{f|r}}{\sum_{f'}(\beta_{f'} + n_{f'|r})}$$

$p(r)$ and $p(f|r)$ are estimated in the M-step:

$$\theta_{doc} = \frac{\alpha + n_{r|doc}}{\sum_{r'}(\alpha + n_{r'|doc})}$$
$$\phi_{rf} = \frac{\beta_f + n_{f|r}}{\sum_{f'}(\beta_{f'} + n_{f'|r})}$$

where $n_{f|r}$ indicates the number of times a feature f is assigned with r.

3.2 Rel-LDA1

Looking at results of Rel-LDA, we find the clusters sometimes are in need of refinement, and we can address this by adding more features. For instance, adding *trigger* features can encourage sparsity over dependency paths. We define trigger words as all the words on the dependency path except stop words. For example, from path "X, based in Y", "base" is extracted as a trigger word. The intuition

for using trigger words is that paths sharing the same set of trigger words should go to one cluster. Adding named entity tag pair can refine the cluster too. For example, a cluster found by Rel-LDA contains "X was born in Y" and "X lives in Y"; but it also contains "X, a company in Y". In this scenario, adding features 'PER-LOC' and 'ORG-LOC' can push the model to split the clusters into two and put the third case into a new cluster.

Hence we propose Rel-LDA1. It is similar to Rel-LDA, except that each tuple is represented with more features. Besides p, s, and d, we introduce trigger words, lexical pattern, POS tag pattern, the named entity pair and the syntactic category pair features for each tuple. Lexical pattern is the word sequence between the two arguments of a tuple and POS tag pattern is the POS tag sequence of the lexical pattern. See Table 1 as an example.

Following typical EM learning(Charniak and Elsner, 2009), we start with a much simpler generative model, expose the model to fewer features first, and iteratively add more features. First, we train a Rel-LDA model, i.e. the model only generates the dependency path, source and destination arguments. After each interval of 10 iterations, we introduce one additional feature. We add the features in the order of trigger, lexical pattern, POS, NER pair, and syntactic pair.

3.3 Type-LDA model

We know that relations can only hold between certain entity types, known as selectional preferences (Ritter et al., 2010; Seaghdha, 2010; Kozareva and Hovy, 2010). Hence we propose Type-LDA model. This model can capture the selectional preferences of relations to their arguments. In the mean time, it clusters tuples into relational clusters, and arguments into different entity clusters. The entity clusters could be interesting in many ways, for example, defining fine-grained entity types and finding new concepts.

We split the features of a tuple into relation level features and entity level features. Relation level features include the dependency path, trigger, lex and POS features; entity level features include the entity mention itself and its named entity tag.

The generative storyline is as follows. At the document level, a multinomial distribution over relations θ_{doc} is drawn from a Dirichlet prior. A document consists of N relation tuples. Each tuple is represented by relation level features (f) and entity level features of source argument (f_s) and destination argument (f_d). For each tuple, a relation r is drawn from Multi(θ_{doc}). The relation level features and two hidden entity types T_1 and T_2 are independently generated from r. Features f_s are generated from T_1 and f_d from T_2. Figure 2 shows the graphical representation of this model.

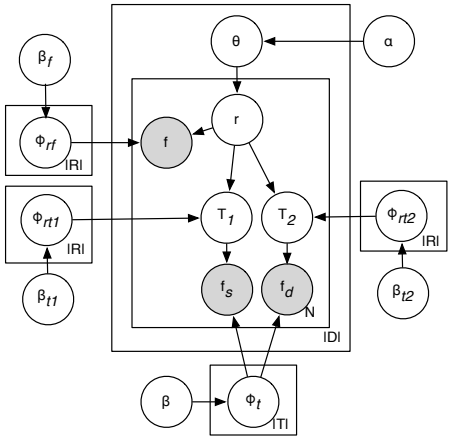

Figure 2: Type-LDA model. Each document consists of N tuples. Each tuple has a set of features, relation level features f and entity level features of source argument f_s and destination argument f_d. Relation level features and two hidden entity types T_1 and T_2 are generated from hidden relation variable r independently. Source entity features are generated from T_1 and destination features are generated from T_2.

At inference time, we sample r, T_1 and T_2 for each tuple. For efficient inference, we first initialize the model without T_1 and T_2, i.e. all the features are generated directly from r. Here the model degenerates to Rel-LDA1. After some iterations, we introduce T_1 and T_2. We sample the relation variable (r) and two mention types variables (T_1,T_2) iteratively for each tuple. We can sample them together, but this is not very efficient. In addition, we found that it does not improve performance.

4 Experiments

Our experiments are carried out on New York Times articles from year 2000 to 2007 (Sandhaus, 2008). We filter out some noisy documents, for example,

obituary content, lists and so on. Obituary articles often contain syntax that diverges from standard newswire text. This leads to parse errors with WSJ-trained parsers and in turn, makes extraction harder. We also filter out documents that contain lists or tables of items (such as books, movies) because this semi-structured information is not the focus of our current work. After filtering we are left with approximately 428K documents. They are preprocessed in several steps. First we employ Stanford tools to tokenize, sentence-split and Part-Of-Speech tag (Toutanova et al., 2003) a document. Next we recognize named entities (Finkel et al., 2005) by labelling tokens with PERSON, ORGANIZATION, LOCATION, MISC and NONE tags. Consecutive tokens which share the same category are assembled into entity mentions. They serve as source and destination arguments of the tuples we seek to model. Finally we parse each sentence of a document using MaltParser (Nivre et al., 2004) and extract dependency paths for each pair of named entity mentions in one sentence.

Following DIRT (Lin and Pantel, 2001), we filter out tuples that do not satisfy the following constraints. First, the path needs to be shorter than 10 edges, since longer paths occur less frequently. Second, the dependency relations in the path should connect two content words, i.e. nouns, verbs, adjectives and adverbs. For example, in phrase 'solve a problem', 'obj(solve, problem)' is kept, while 'det(problem, a)' is discarded. Finally, the dependency labels on the path must not be: 'conj', 'ccomp', 'parataxis', 'xcomp', 'pcomp', 'advcl', 'punct', and 'infmod'. This selection is based on the observation that most of the times the corresponding dependency relations do not explicitly state a relation between two candidate arguments.

After all entity mentions are generated and paths are extracted, we have nearly 2.5M tuples. After clustering (inference), each of these tuple will belong to one cluster/relation and is associated with its clusterID.

We experimented with the number of clusters and find that in a range of 50-200 the performance does not vary significantly with different numbers. In our experiments, we cluster the tuples into 100 relation clusters for all three models. For Type-LDA model, we use 50 entity clusters.

We evaluate our models in two ways. The first aims at measuring the clustering quality by mapping clusters to Freebase relations. The second seeks to assess the utility of our predicted clusters as features for relation extraction.

4.1 Relations discovered by different models

Looking closely at the clusters we predict, we find that some of them can be mapped to Freebase relations. We discover clusters that roughly correspond to the *parentCom* (parent company relation), *filmDirector*, *authorOf*, *comBase* (base of a company relation) and *dieIn* relations in Freebase. We treat Freebase annotations as ground truth and measure recall. We count each tuple in a cluster as true positive if Freebase states the corresponding relation between its argument pair. We find that precision numbers against Freebase are low, below 10%. However, these numbers are not reliable mainly because many correct instances found by our models are missing in Freebase. One reason why our predictions are missing in Freebase is coreference. For example, we predict *parentCom* relation between 'Linksys' and 'Cisco', while Freebase only considers 'Cisco Systems, Inc.' as the parent company of 'Linksys'. It does not corefer 'Cisco' to 'Cisco Systems, Inc.'. Incorporating coreference in our model may fix this problem and is a focus of future work. Instead of measuring precision against Freebase, we ask humans to label 50 instances for each cluster and report precision according to this annotated data. Table 3 shows the scores.

We can see that in most cases Rel-LDA1 and Type-LDA substantially outperform the Rel-LDA model. This is due to the fact that both models can exploit more features to make clustering decisions. For example, in Rel-LDA1 model, the NER pair feature restricts the entity types the two arguments can take.

In the following, we take *parentCom* relation as an example to analyze the behaviors of different models. Rel-LDA includes spurious instances such as 'A is the chief executive of B', while Rel-LDA1 has fewer such instances due to the NER pair feature. Similarly, by explicitly modeling entity type constraints, Type-LDA makes fewer such errors. All our models make mistakes when sentences have coordination structures on which the parser has failed.

Rel.	Sys.	Rec.	Prec.
parentCom	Rel-LDA	51.4	76.0
	Rel-LDA1	49.5	**78.0**
	Type-LDA	**55.3**	72.0
filmDirector	Rel-LDA	42.5	32.0
	Rel-LDA1	70.5	**40.0**
	Type-LDA	**74.2**	26.0
comBase	Rel-LDA	31.5	12.0
	Rel-LDA1	54.2	22.0
	Type-LDA	**57.1**	**30.0**
authorOf	Rel-LDA	25.2	**84.0**
	Rel-LDA1	**46.9**	86.0
	Type-LDA	20.2	68.0
dieIn	Rel-LDA	26.5	34.0
	Rel-LDA1	**55.9**	**40.0**
	Type-LDA	50.2	28.0

Table 3: Clustering quality evaluation (%), Rec. is measured against Freebase, Prec. is measured according to human annotators

For example, when a sentence has the following pattern "The winners are A, a part of B; C, a part of D; E, a part of F", our models may predict *parentCom*(A,F), because the parser connects A with F via the pattern 'a part of'.

Some clusters found by our models cannot be mapped to Freebase relations. Consider the Freebase relation *worksFor* as one example. This relation subsumes all types of employment relationships, irrespective of the role the employee plays for the employer. By contrast, our models discover clusters such as *leaderOf*, *editorOf* that correspond to more specific roles an employee can have. We show some example relations in Table 4. In the table, the 2nd row shows a cluster of employees of news media companies; the 3rd row shows leaders of companies; the last one shows birth and death places of persons. We can see that the last cluster is noisy since we do not handle antonyms in our models. The arguments of the clusters have noise too. For example, 'New York' occurs as a destination argument in the 2nd cluster. This is because 'New York' has high frequency in the corpus and it brings noise to the clustering results. In Table 5 some entity clusters found by Type-LDA are shown. We find different types of companies, such as financial companies and news companies. We also find subclasses of *person*, for example, *reviewer* and *politician*, because these different entity classes participate in different relations. The last cluster shown in the table is a mixture of news companies and government agencies. This may be because this entity cluster is affected by many relations.

4.2 Distant Supervision based Relation Extraction

Our generative models detect clusters of dependency paths and their arguments. Such clusters are interesting in their own right, but we claim that they can also be used to help a supervised relation extractor. We validate this hypothesis in the context of relation extraction with distant supervision using predicted clusters as features.

Following previous work (Mintz et al., 2009), we use Freebase as our distant supervision source, and align related entity pairs to the New York Times articles discussed earlier. Our training and test instances are pairs of entities for which both arguments appear in at least one sentence together. Features of each instance are extracted from all sentences in which both entities appear together. The gold label for each instance comes from Freebase. If a pair of entities is not related according to Freebase, we consider it a negative example. Note that this tends to create some amount of noise: some pairs may be related, but their relationships are not yet covered in Freebase.

After filtering out relations with fewer than 10 instances we have 65 relations and an additional "O" label for unrelated pairs of entities. We call related instances positive examples and unrelated instances negative examples.

We train supervised classifiers using maximum entropy. The baseline classifier employs features that Mintz et al. (2009) used. To extract features from the generative models we proceed as follows. For each pair of entities, we collect all tuples associated with it. For each of these tuples we extract its clusterID, and use this ID as a binary feature.

The baseline system without generative model features is called *Distant*. The classifiers with additional features from generative models are named after the generative models. Thus we have Rel-LDA, Rel-LDA1 and Type-LDA classifiers. We compare

Source	New York, Euro RSCG Worldwide, BBDO Worldwide, American, DDB Worldwide
Path	X, a part of Y; X, a unit of Y; X unit of Y; X, a division of Y; X is a part of Y
Dest	Omnicom Group, Interpublic Group of Companies, WPP Group, Publicis Groupe
Source	Supreme Court, Anna Wintour, William Kristol, Bill Keller, Charles McGrath
Path	X, an editor of Y; X, a publisher of Y; X, an editor at Y; X, an editor in chief of Y; X is an editor of Y;
Dest	The Times, The New York Times, Vogue, Vanity Fair, New York
Source	Kenneth L. Lay, L. Dennis Kozlowski, Bernard J. Ebbers, Thomas R. Suozzi, Bill Gates
Path	X, the executive of Y; X, Y's executive; X, Y executive; X, the chairman of Y; X, Y's chairman
Dest	Enron, Microsoft, WorldCom, Citigroup, Nassau County
Source	Paul J. Browne, John McArdle, Tom Cocola, Claire Buchan, Steve Schmidt
Path	X, a spokesman for Y; X, a spokeswoman for Y; X, Y spokesman; X, Y spokeswoman; X, a commissioner of Y
Dest	White House, Justice Department, Pentagon, United States, State Department
Source	United Nations, Microsoft, Intel, Internet, M. D. Anderson
Path	X, based in Y; X, which is based in Y; X, a company in Y; X, a company based in Y; X, a consultant in Y
Dest	New York, Washington, Manhattan, Chicago, London
Source	Army, Shiite, Navy, John, David
Path	X was born in Y; X die at home in Y; X die in Y; X, son of Y; X die at Y
Dest	Manhattan, World War II, Brooklyn, Los Angeles, New York

Table 4: The path, source and destination arguments of some relations found by Rel-LDA1.

Company	Microsoft, Enron, NBC, CBS, Disney
FinanceCom	Merrill Lynch, Morgan Stanley, Goldman Sachs, Lehman Brothers, Credit Suisse First Boston
News	Notebook, New Yorker, Vogue, Vanity Fair, Newsweek
SportsTeam	Yankees, Mets, Giants, Knicks, Jets
University	University of California, Harvard, Columbia University, New York University, University of Penn.
Art Reviewer	Stephen Holden, Ken Johnson, Roberta Smith, Anthony Tommasini, Grace Glueck
Games	World Series, Olympic, World Cup, Super Bowl, Olympics
Politician	Eliot Spitzer, Ari Fleischer, Kofi Annan, Scott McClellan, Karl Rove
Gov. Agency	Congress, European Union, NATO, Federal Reserve, United States Court of Appeals
News/Agency	The New York Times, The Times, Supreme Court, Security Council, Book Review

Table 5: The entity clusters found by Type-LDA

these against Distant and the DIRT database. For the latter we parse our data using Minipar (Lin, 1998) and extract dependency paths between pairs of named entity mentions. For each path, the top 3 similar paths are extracted from DIRT database. The Minipar path and the similar paths are used as additional features.

For held-out evaluation, we construct the training data from half of the positive examples and half of the negative examples. The remaining examples are used as test data. Note that the number of negative instances is more than 10 times larger than the number of positive instances. At test time, we rank the predictions by the conditional probabilities obtained from the Maximum Entropy classifier. We report precision of top ranked 50 instances for each relation in table 6. From the table we can see that all systems using additional features outperform the Distant system. In average, our best model achieves 4.1% improvement over the distant supervision baseline, 12% error reduction. The precision of *bornIn* is low because in most cases we predict *bornIn* instances as *liveIn*.

We expect systems using generative model features to have higher recall than the baseline. This is difficult to measure, but precision in the high recall area is a signal. We look at top ranked 1000 instances of each system and show the precision in the last row of the table. We can see that our best model Type-LDA outperforms the distant supervision baseline by 4.5%.

Why do generative model features help to im-

Relation	Dist	Rel	Rel1	Type	DIRT
worksFor	80.0	**92.0**	86.0	90.0	84.0
authorOf	98.0	98.0	98.0	98.0	98.0
containedBy	92.0	**96.0**	**96.0**	92.0	**96.0**
bornIn	16.0	18.0	22.0	**24.0**	10.0
dieIn	28.0	**30.0**	28.0	24.0	24.0
liveIn	50.0	52.0	54.0	54.0	**56.0**
nationality	92.0	**94.0**	90.0	90.0	**94.0**
parentCom	94.0	**96.0**	**96.0**	**96.0**	90.0
founder	65.2	**76.3**	61.2	64.0	68.3
parent	52.0	**54.0**	50.0	52.0	52.0
filmDirector	54.0	60.0	60.0	**64.0**	62.0
Avg	65.6	**69.7**	67.4	68.0	66.8
Prec@1K	82.8	85.8	85.3	**87.3**	82.8

Table 6: Precision (%) of some frequent relations

prove relation extraction? One reason is that generative models can transfer information from known patterns to unseen patterns. For example, given "Sidney Mintz, the great food anthropologist at Johns Hopkins University", we want to predict the relation between 'Sidney Mintz' and 'Johns Hopkins University'. The distant supervision system incorrectly predicts the pair as 'O' since it has not seen the path 'X, the anthropologist at Y' in the training data. By contrast, Rel-LDA can predict this pair correctly as *worksFor* because the dependency path of this pair is in a cluster which contains the path 'X, a professor at Y'.

In addition to held-out evaluation we also carry out manual evaluation. To this end, we use all the positive examples and randomly select five times the number of positive examples as negative examples to train a classifier. The remaining negative examples are candidate instances. We rank the predicted instances according to their classification scores. For each relation, we ask human annotators to judge its top ranked 50 instances.

Table 7 lists the manual evaluation results for some frequent relations. We also list how many instances are found for each relation. For almost all the relations, systems using generative model features find more instances. In terms of precision, our models perform comparatively to the baseline, even better for some relations.

We also notice that clustering quality is not consistent with distant supervision performance. Rel-LDA1 can find better clusters than Rel-LDA but it has lower precision in held-out evaluation. Type-LDA underperforms Rel-LDA in average precision but it gets higher precision in a higher recall area, i.e. precision at 1K. One possible reason for the inconsistency is that the baseline distant supervision system already employs features that are used in Rel-LDA1. Another reason may be that the clusters do not overlap with Freebase relations very well, see section 4.1.

4.3 Comparing against USP

We also try to compare against USP (Poon and Domingos, 2008). Due to memory requirements of USP, we are only able to run it on a smaller data set consisting of 1,000 NYT documents; this is three times the amount of data Poon and Domingos (2008) used to train USP.[2] For distant supervision based relation extraction, we only match about 500 Freebase instances to this small data set.

USP provides a parse tree for each sentence and for each mention pair we can extract a path from the tree. Since USP provides clusters of words and phrases, we use the USP clusterID associated with the words on the path as binary features in the classifier.

All models are less accurate when trained on this smaller dataset; we can do as well as USP does, even a little better. USP achieves 8.6% in F1, Rel-LDA 8.7%, Rel-LDA1 10.3%, Type-LDA 8.9% and Distant 10.3%. Of course, given larger datasets, the performance of Rel-LDA, Rel-LDA1, and Type-LDA improves considerably. In summary, comparing against USP, our approach scales much more easily to large data.

5 Related Work

Many approaches have been explored in relation extraction, including bootstrapping, supervised classification, distant supervision, and unsupervised approaches.

Bootstrapping employs a few labeled examples for each relation, iteratively extracts patterns from the labeled seeds, and uses the patterns to extract

[2]Using the publicly released USP code, training a model with 1,000 documents resulted in about 45 gigabytes of heap space in the JVM.

Relation	Top 50 (%)			#Instances		
	Dist	Rel	Type	Dist	Rel	Type
worksFor	**100.0**	**100.0**	**100.0**	314	**349**	**349**
authorOf	94.0	94.0	**96.0**	185	208	**229**
containedBy	**98.0**	**98.0**	**98.0**	670	714	**804**
bornIn	82.6	**88.2**	88.0	46	36	**56**
dieIn	**100.0**	**100.0**	**100.0**	167	176	**231**
liveIn	**98.0**	**98.0**	94.0	77	86	**109**
nationality	78.0	**82.0**	76.0	84	92	**114**
parentCom	79.2	77.4	**85.7**	24	**31**	28
founder	**80.0**	**80.0**	50.0	5	5	**14**
parent	**97.0**	92.3	94.7	33	**39**	38
filmDirector	92.6	96.9	**97.1**	27	32	**34**

Table 7: Manual evaluation, Precision and recall of some frequent relations

more seeds (Brin, 1998). This approach may suffer from low recall since the patterns can be too specific.

Supervised learning can discover more general patterns (Kambhatla, 2004; Culotta and Sorensen, 2004). However, this approach requires labeled data, and most work only carry out experiments on small data set.

Distant supervision for relation extraction requires no labeled data. The approach takes some existing knowledge base as supervision source, matches its relational instances against the text corpus to build the training data, and extracts new instances using the trained classifiers (Mintz et al., 2009; Bunescu and Mooney, 2007; Riedel et al., 2010; Yao et al., 2010).

All these approaches can not discover new relations and classify instances which do not belong to any of the predefined relations. Other past work has explored inducing relations using unsupervised approaches.

For example, DIRT (Lin and Pantel, 2001) aims to discover different representations of the same semantic relation, i.e. similar dependency paths. They employ the distributional similarity based approach while we use generative models. Both DIRT and our approach take advantage of the arguments of dependency paths to find semantic relations. Moreover, our approach can cluster the arguments into different types.

Unsupervised semantic parsing (USP) (Poon and Domingos, 2008) discovers relations by merging predicates which have similar meanings; it proceeds to recursively cluster dependency tree fragments (or "parts") to best explain the observed sentence. It is not focused on capturing any particular kind of relation between sentence constituents, but to capture repeated patterns. Our approach differs in that we are focused on capturing a narrow range of binary relations between named entities; some of our models (see Section 3) utilize entity type information to constraint relation type induction. Also, our models are built to be scalable and trained on a very large corpus. In addition, we use a distant supervision framework for evaluation.

Relation duality (Bollegala et al., 2010) employs co-clustering to find clusters of entity pairs and patterns. They identify each cluster of entity pairs as a relation by selecting representative patterns for that relation. This approach is related to our models, however, it does not identify any entity clusters.

Generative probabilistic models are widely employed in relation extraction. For example, they are used for in-domain relation discovery while incorporating constraints via posterior regularization (Chen et al., 2011). We are focusing on open domain relation discovery. Generative models are also applied to selectional preference discovery (Ritter et al., 2010; Seaghdha, 2010). In this scenario, the authors assume relation labels are given while we automatically discover relations. Generative models are also used in unsupervised coreference (Haghighi and Klein, 2010).

Clustering is also employed in relation extraction. Hasegawa et al. (2004) cluster pairs of named entities according to the similarity of context words intervening between them. Their approach is not probabilistic. Researchers also use topic models to perform dimension reduction on features when they cluster relations (Hachey, 2009). However, they do not explicitly model entity types.

Open information extraction aims to discover relations independent of specific domains and relations (Banko et al., 2007; Banko and Etzioni, 2008). A self-learner is employed to extract relation instances but the systems do not cluster the instances into relations. Yates and Etzioni (2009) present RESOLVER for discovering relational synonyms as a post processing step. Our approach integrates entity and relation discovery in a probabilistic model.

6 Conclusion

We have presented an unsupervised probabilistic generative approach to relation extraction between two named entities. Our proposed models exploit entity type constraints within a relation as well as features on the dependency path between entity mentions to cluster equivalent textual expressions. We demonstrate the effectiveness of this approach by comparing induced relation clusters against a large knowledge base. We also show that using clusters of our models as features in distant supervised framework yields 12% error reduction in precision over a weakly supervised baseline and outperforms other state-of-the art relation extraction techniques.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and the University of Massachusetts gratefully acknowledges the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181, ITR#1, and NSF MALLET. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of the DARPA, AFRL, or the US government. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

Michele Banko and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. In *Proceedings of ACL-08: HLT*.

Michele Banko, Michael J Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of IJCAI2007*.

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

Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2010. Relational duality: Unsupervised extraction of semantic relations between entities on the web. In *Proceedings of WWW*.

Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In *Proc. of WebDB Workshop at 6th International Conference on Extending Database Technology*.

Razvan C. Bunescu and Raymond J. Mooney. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45rd Annual Meeting of the Association for Computational Linguistics (ACL '07)*.

Eugene Charniak and Micha Elsner. 2009. Em works for pronoun anaphora resolution. In *Proceedings of ACL*.

Harr Chen, Edward Benson, Tahira Naseem, and Regina Barzilay. 2011. In-domain relation discovery with meta-constraints via posterior regularization. In *Proceedings of ACL*.

Aron Culotta and Jeffery Sorensen. 2004. Dependency tree kernels for relation extraction. In *42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain.

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 (ACL '05)*, pages 363–370, June.

Benjamin Hachey. 2009. *Towards Generic Relation Extraction*. Ph.D. thesis, University of Edinburgh.

Aria Haghighi and Dan Klein. 2010. Coreference resolution in a modular, entity-centered model. In *Proceedings of HLT-NAACL*.

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *ACL*.

Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of ACL*.

Zornitsa Kozareva and Eduard Hovy. 2010. Learning arguments and supertypes of semantic relations using recursive patterns. In *Proceedings of ACL 10*.

Dekang Lin and Patrick Pantel. 2001. Dirt - discovery of inference rules from text. In *Proceedings of KDD*.

Dekang Lin. 1998. Dependency-based evaluation of minipar. In *Proceedings of the Workshop on the Evaluation of Parsing Systems*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL-IJCNLP*.

J. Nivre, J. Hall, and J. Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of CoNLL*, pages 49–56.

Hoifung Poon and Pedro Domingos. 2008. Unsupervised semantic parsing. In *Proceedings of the Conference on Empirical methods in natural language processing (EMNLP)*.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL*.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD '10)*.

Alan Ritter, Mausam, and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *Proceedings of ACL10*.

Dan Roth and Wen tau Yih. 2002. Probabilistic reasoning for entity and relation recognition. In *Proceedings of Coling*.

Evan Sandhaus, 2008. *The New York Times Annotated Corpus*. Linguistic Data Consortium, Philadelphia.

Diarmuid O Seaghdha. 2010. Latent variable models of selectional preference. In *Proceedings of ACL 10*.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT-NAACL*, pages 252–259.

Limin Yao, Sebastian Riedel, and Andrew McCallum. 2010. Collective cross-document relation extraction without labelled data. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1013–1023, Cambridge, MA, October. Association for Computational Linguistics.

Alexander Yates and Oren Etzioni. 2009. Unsupervised methods for determining object and relation synonyms on the web. *Journal of Artificial Intelligence Research*, 34:255–296.