Active Learning for Relation Type Extension with Local and Global Data Views

Ang Sun*
New York University
719 Broadway, Room 706
New York, NY 10003, USA
asun@cs.nyu.edu

Ralph Grishman
New York University
715 Broadway, Room 703
New York, NY 10003, USA
grishman@cs.nyu.edu

ABSTRACT

Relation extraction is the process of identifying instances of specified types of semantic relations in text; relation type extension involves extending a relation extraction system to recognize a new type of relation. We present *LGCo-Testing*, an active learning system for relation type extension based on *local* and *global* views of relation instances. *Locally*, we extract features from the sentence that contains the instance. *Globally*, we measure the distributional similarity between instances from a 2 billion token corpus. Evaluation on the ACE 2004 corpus shows that *LGCo-Testing* can reduce annotation cost by 97% while maintaining the performance level of supervised learning.

Categories and Subject Descriptors

H.3 [**INFORMATION STORAGE AND RETRIEVAL**]: Miscellaneous; I.2.7 [ARTIFICIAL INTELLIGENCE]: Natural Language Processing – *Language parsing and understanding*

Keywords

Information extraction, relation extraction, active learning, co-training, co-testing, distributional similarity

1. Introduction

Relation extraction is the process of identifying instances of specified types of semantic relations in text; relation type extension involves extending a relation extraction system to recognize a new type of relation. Relation extraction aims to connect related entities in text and label them with the right semantic relationship. For example, a relation extraction system needs to detect an *Employment* relation between the entities *He* and *Arkansas* in the sentence *He was the governor of Arkansas*. The task of relation type extension is to adapt an existing relation extraction system to be able to extract a new type of relation, often called the *target relation*, preferably in a fast, cheap and accurate way.

A supervised approach can tackle this problem in an accurate way. But it is slow and expensive as it relies on human annotation of a large quantity of examples. Semi-supervised learning, in contrast, does not require much human effort by automatically bootstrapping a system for the *target relation* from only a few labeled examples and a large unlabeled corpus. However, a large gap still exists between the performance of semi-supervised and supervised systems. Moreover, their performance largely depends on the choice of seeds [10, 15].

Another attractive alternative is active learning, which reduces annotation cost by requesting labels of only the most informative examples while maintaining high learning performance. It is also shown to be robust to the choice of the seeds [7]. Specifically, we focus on relation type extension with *co-testing* [12], an active learning approach in the *co-training* [1] setting. It minimizes human annotation effort by building two classifiers based on two different views of the data and asking for human labels only for contention data points, points on which the two classifiers disagree about their labels. The key to the success of *co-testing* is to find a natural way of splitting a data point into two views that are *uncorrelated* and *compatible* (each view is sufficient in labeling the data point).

To date, there is limited work on applying *co-testing* to relation type extension. The main difficulty, as we believe, is the lack of a natural way of partitioning the data into two *uncorrelated* and *compatible* views. Unlike named entity classification where one can rely on either the name string itself (*Arkansas*) or the context (*was the governor of* <>) to determine the type of the named entity [4, 7], the type of a relation is mostly determined by the context in which the two entities appear. For example, it is not possible to decide the type of relation between *He* and *Arkansas* without the local context *was the governor of*. If the context was *traveled to*, then the relation of the two entities would change entirely. Thus it is not desirable to separate the entities from their context to establish two views. Instead, we treat them together as a single view, the local view.

Motivated by the idea of distributional similarity, we move beyond the local view to interpret the relation between two entities. Specifically, we compute from a 2 billion token corpus the distributional similarities between relational phrases. We take these similarities as the global view upon which we build a classifier which classifies new examples based on the *k*-nearest neighbor algorithm. For example, if the phrase *arrived in* is more similar to *traveled to* than to *was the governor of*, the global view classifier classifies entities

*Present address: Intelius Inc. 500 108th Ave NE Suite 2200 Bellevue, WA 98004. asun@intelius.com

CIKM'12, October 29-November 2, 2012, Maui, HI, USA.

connected by *arrived in* as the same relation as those connected by *traveled to*. Armed with this global view classifier and a classifier trained with features extracted from the local view, applying *co-testing* to relation type extension becomes feasible.

The main contributions of active learning with local and global views are: it indeed introduces two *uncorrelated* and *compatible* views for relation extraction. It provides substantial reduction of annotation effort as compared to various baselines based on an extensive experimental study. Furthermore, it leads to faster convergence of learning.

The next section introduces our task. Sections 3 and 4 describe the local and global view classifiers in detail. We present *LGCo-Testing* and baseline systems in Section 5 and 6, and evaluate them in Section 7. We discuss related work in Section 8 and conclude in Section 9.

2. Task Definition

We choose to work on the well-defined relation extraction task of the Automatic Content Extraction[1] (ACE) program in 2004, mostly driven by the purpose of system comparison as many published results on this data set are available. A relation is defined over a pair of entities within a single sentence. ACE 2004 defined 7 major relation types. Some examples are shown in Table 1. Following previous work, we only deal with relation mentions.

Table 1: ACE relation examples from the annotation guideline[2]. Heads of entity mentions are marked.

Type	Example
EMP-ORG	*the CEO of Microsoft*
PHYS	*a military base in Germany*
GPE-AFF	*U.S. businessman*
PER-SOC	*his ailing father*
DISC	*each of whom*
ART	*US helicopters*
OTHER-AFF	*Cuban-American people*

We consider two training situations: one where we are defining a new relation for a previously unannotated corpus; we call this the *binary setting*; the other where we are adding a new relation to a corpus previously annotated with *n* relations, where the *n+1* relations are mutually exclusive; we call this the *multi-class setting*. We refer to the new relation as the *target relation*. We use the ACE 2004 corpus, which is annotated with 7 different relation types, to experiment with both tasks. For the *binary setting*, we pick one of the ACE relations as the *target relation* and pretend we don't know anything about the other relations. For the *multi-class* setting, we again pick one of the ACE relations as the *target relation* and treat the remaining 6 types as *auxiliary relations* which we already know about and can use to aid the learning of the new type.

3. The Local View Classifier

There are two common learning approaches for building a classifier based on the local view: feature-based [8, 6, 20] and kernel-based [2, 16, 17, 19, 21]. As we want to compare *LGCo-Testing* with *co-testing* based on a feature split at the local level, we choose to build a feature-based local classifier.

Given a relation instance $x = (s, e_i, e_j)$, where e_i and e_j are a pair of entities and s is the sentence containing the pair, the local classifier starts with multiple level analyses of the sentence such as tokenization, syntactic parsing, and dependency parsing. It then extracts a feature vector v which contains a variety of lexical, syntactic and semantic features for each relation instance. Our features are cherrypicked from previous feature based systems. Table 2 shows the feature set with examples.

After feature engineering, the local classifier applies machine learning algorithms to learn a function which can estimate the conditional probability $p(c|v)$, the probability of the type c given the feature vector v of the instance x. We used maximum entropy (MaxEnt) to build a binary classifier (for the *binary setting*) and a multi-class classifier (for the *multi-class setting*) because the training is fast, which is crucial for active learning as it is not desirable to keep the annotator waiting because of slow training.

4. The Global View Classifier

Building a classifier based on the global view involves three stages of process: extracting relational phrases, computing distributional similarities, and building the classifier based on similarities. We describe these stages in detail below.

Extracting relational phrases. Given a relation instance $x = (s, e_i, e_j)$ and assuming that e_i appears before e_j, we represent it as a relational phrase p_x, which is defined as the n-gram that spans the head[3] of e_i and that of e_j. Formally, $p_x = [head\_e_i, head\_e_j]$. For example, we extract *Clinton traveled to the Irish border* as the phrase for the example in Table 2. As our goal is to collect the tokens before and after a phrase as features to capture the similarity between phrases and long phrases are too sparse to be useful, we instead use the definition $p_x = (e_i, e_j)$ (tokens between e_i and e_j) when the phrase contains more than 5 tokens. Thus for the example in Table 2, because the previously extracted phrase contains 6 tokens, we will instead use the phrase "*traveled to*" to represent that instance.

Computing distributional similarities. We first compile our 2 billion token text corpus to a database of 7-grams and then form 7-gram queries to extract features for a phrase. Example queries for the phrase *traveled to* are shown in Table 3.

[1] Task definition: http://www.itl.nist.gov/iad/894.01/tests/ace/

ACE guidelines: http://projects.ldc.upenn.edu/ace/

[2] http://projects.ldc.upenn.edu/ace/docs/EnglishRDCV4-3-2.PDF

[3] The last token of a noun group.

Table 2: Sample features for "<e_i>President Clinton</e_i> traveled to <e_j>the Irish border</e_j> for an …"

Level	Type	Description	Value
Entity	*ET*	Entity types	*ET1=PERSON; ET2= LOCATION*
	ET12	Combination of ET1 and ET2	*ET12=PERSON--LOCATION*
	ML	Mention levels	*ML1=NAME; ML2= NOMINAL*
	ML12	Combination of ML1 and ML2	*ML12=NAME--NOMINAL*
	HE	Heads of entities	*HE1=Clinton; HE2= border*
	HE12	Combination of HE1 and HE2	*Clinton--border*
	BagWE	Bag of words of entities	*{President, Clinton}{the, Irish, border}*
Sequence	*WBE1*	Words before entity 1	*{NIL}*
	WB	Words between	*{traveled, to}*
	WAE2	Words after entity 2	*{for, an}*
	NUMWB	# words between	*2*
	TPatternET	Token pattern coupled with entity types	*PERSON_traveled_to_LOCATION*
Syntactic Parsing	*PTP*	Path of phrase labels connecting E1 and E2 in the parsing tree	*NP--VP--PP*
	PTPH	PTP augmented with the head word of the top phrase in the path	*NP--S--traveled--VP--PP*
	CPHBE1	Chunk heads before E1	*{NIL}*
	CPHB	Chunk heads between	*{traveled, to}*
	CPHAE2	Chunk heads after E2	*{for, ceremony}*
	CPP	Path of phrase labels connecting the two entities in the chunking	*NP--VP/S--PP--NP*
	CPPH	CPP augmented with head words	*NP-Clinton-VP/S-traveled-PP-to-NP-border*
	CPatternET	Chunk head pattern with entity types	*PERSON_traveled_to_LOCATION*
Dependency Parsing	*DPathET*	Shortest dependency path connecting the two entities coupled with entity types	*PER_nsubj'_traveled_prep_to_LOC*
	ET1DW1	Entity type and the dependent word for E1	*PERSON--traveled*
	ET2DW2	Entity type and the dependent word for E2	*LOCATION--to*
	H1DW1	The head and the dependent word for E1	*Clinton--traveled*
	H2DW2	Head word and the dependent word for E2	*border--to*
	ET12Same NP/PP/VP	Whether E1 and E2 included in the same NP/PP/VP	*false/false/true*

Table 3: 7-gram queries for *traveled to.*

* * * * * traveled to	traveled to * * * * *
* * * * traveled to *	* traveled to * * * *
* * * traveled to * *	* * traveled to * * *

We then collect the tokens that could instantiate the wild cards in the queries as features. Note that tokens are coupled with their positions. For example, if the matched 7-gram is *President Clinton traveled to the Irish border*, we will extract from it the following five features: *President_-2*, *Clinton_-1*, *the_+1*, *Irish_+2* and *border_+3*.

Each phrase *P* is represented as a feature vector of contextual tokens. To weight the importance of each feature *f*, we first collect its counts, and then compute an analogue of *tf-idf*: *tf* as the number of corpus instances of *P* having feature *f* divided by the number of instances of *P*; *idf* as the total number of phrases in the corpus divided by the number of phrases with at least one instance with feature *f*. Now the token feature vector is transformed into a *tf-idf* feature vector. We compute the similarity between two vectors using *Cosine* similarity. The most similar phrases of *traveled to* and *his family* are shown in Table 4.

Table 4: Sample of similar phrases.

traveled to		*his family*	
Phrase	***Sim.***	***Phrase***	***Sim.***
visited	0.779	his staff	0.792
arrived in	0.763	his brother	0.789
worked in	0.751	his friends	0.780
lived in	0.719	his children	0.769
served in	0.686	their families	0.753
consulted with	0.672	his teammates	0.746
played for	0.670	his wife	0.725

Building the classifier. We build the global view classifier based on the *k*-nearest neighbor idea, classifying an unlabeled example based on closest labeled examples. The similarity between an unlabeled instance u and a labeled instance l is measured by the similarity between their phrases, p_u and p_l. Note that we also incorporate the entity type constraints into the similarity computation. The similarity is defined to be zero if the entity types of u and l do not match. The similarity between u and a relation type c, $sim(u, c)$, is estimated by the similarity between u and its k closest instances in the labeled instance set of c (we take the averaged similarity if $k > 1$; we will report results with $k=3$ as it works slightly better than 1, 2, 4 and 5). Let $h(u)$ be the classification function, we define it as follows:

$$h(u) = argmax_c sim(u, c)$$

5. LGCo-Testing

We first introduce a general *co-testing* procedure, then describe the details of the proposed *LGCo-Testing*.

Let D_U denote unlabeled data, and D_L denote labeled data, the *co-testing* procedure repeats the following steps until it converges:

1. Train two classifiers h_1 and h_2 based on two data views with D_L
2. Label D_U with h_1 and h_2 and build a contention set S
3. Select $S' \subseteq S$ based on *informativeness* and request human labels
4. Update: $D_L = D_L \cup S'$ and $D_U = D_U \backslash S'$

Initialization. Initialization first concerns the choice of the seeds. For the *multi-class setting*, it also needs to effectively introduce the instances of *auxiliary* relations.

For the choice of the seeds, as we are doing simulated experiments on the ACE corpus, we take a random selection strategy and hope multiple runs of our experiments can approximate what will actually happen in real world active learning. Moreover, it was empirically found that active learning is able to rectify itself from bad seeds [7]. In all experiments for both the *binary* and the *multi-class* settings, we use as seeds 5 randomly selected *target relation* instances and 5 randomly selected *non-relation* instances (entity pairs in a sentence not connected by an ACE relation).

For the *multi-class* setting, we use a stratified strategy to introduce the *auxiliary* relation instances: the number of selected instances of a type is proportional to that of the total number of instances in the labeled data. Since we use the annotated ACE corpus and assume full knowledge of the auxiliary relations over the corpus, we can accurately estimate their distribution.

We also make the assumption that our *target relation* is as important as the most frequent *auxiliary* relation and select these two types equally. For example, assuming that we only have two *auxiliary* types with 100 and 20 labeled instances respectively, we will randomly select 5 instances for the first type and 1 instance for the second type, given that we initialized our active learning with 5 *target relation* seeds. We also experimented with several other ways of introducing the *auxiliary* relation instances and none of them were as effective as the stratified strategy. For one example, using all the *auxiliary* instances to train the initial classifiers unfortunately generates an extremely unbalanced class distribution and tends to be biased towards the *auxiliary* relations. For another, selecting the same number of instances for the *target relation* type and all the *auxiliary* types does not take full advantage of the class distribution of the *auxiliary* types, which can be estimated with the labeled ACE data set.

Informativeness measurement. It is straightforward to get the *hard* labels of an instance from both the local and global view classifiers. As the local classifier uses MaxEnt which is essentially logistic regression, we take the class with the highest probability as the *hard* label. The *hard* label of the global classifier is the relation type to which the instance is most similar. As long as the two classifiers disagree about an instance's *hard* label and one of the labels is our *target relation*, we add it to the contention set.

Quantifying the disagreement between the two classifiers is not as straightforward as getting the *hard* labels because the local classifier produces a probability distribution over the relation types while the global classifier produces a similarity distribution. So we first use the following formula to transform similarities to probabilities.

$$p(c|u) = \frac{exp\ (sim(u, c))}{\sum_i exp\ (sim(u, c_i))}$$

Here u is an instance that needs to be labeled, c is a specific relation type, and $sim(u, c_i)$ is the similarity between u and one of the relation types c_i.

We then use *KL-divergence* to quantify the degree of deviation between the two probability distributions. *KL-divergence* measures the divergence between two probability distributions p and q over the same event space χ:

$$D(p||q) = \sum_{x \in \chi} p(x) \log \frac{p(x)}{q(x)}$$

It is non-negative. It is zero for identical distributions and achieves its maximum value when distributions are peaked and prefer different labels. We rank the contention instances by descending order of *KL-divergence* and pick the top 5 instances to request human labels during a single iteration.

It is worth mentioning that, for each iteration in the *multi-class* setting, *auxiliary* instances are introduced using the stratified strategy as in the initialization step.

Convergence detection. We stop *LGCo-Testing* when we could not find contention instances.

6. Baselines

We compare our approach to a variety of baselines, including six active learning baselines, one supervised system and one semi-supervised system. We present the details of active learning baselines below, and refer the reader to the experiment section to learn more about other baselines.

SPCo-Testing. One of the many competitive active learning approaches is to build two classifiers based on a feature split at the local level. As reported by [6], either the sequence features or the parsing features are generally sufficient to achieve state-of-the-art performance for relation extraction. So we build one classifier based on the sequence view and the other based on the parsing view. More precisely, one classifier is built with the feature set based on *{entity, sequence}* and the other based on *{entity, syntactic parsing, dependency parsing}*.

We build these two classifiers with MaxEnt. The initialization is the same as in *LGCo-Testing*. *KL-divergence* is used to quantify the disagreement between the two probability distributions returned by the two MaxEnt classifiers. Contention points are ranked in descending order of *KL-divergence* and the top 5 are used to query the annotator in one iteration. Like *LGCo-Testing*, *SPCo-Testing* stops when the contention set is empty.

UncertaintyAL. This is an uncertainty-based active learning baseline. We build a single MaxEnt classifier based on the full feature set in Table 2 at the local level. It uses the same initialization as in *LGCo-Testing*. Informativeness measurement is based on *uncertainty*, which is approximated by the entropy $h(p)$ of the probability distribution of the MaxEnt classifier over all the relation types c_i.

$$h(p) = -\sum_i p(c_i) \log p(c_i)$$

It is also non-negative. It is zero when one relation type is predicted with a probability of 1. It attains its maximum value when the distribution is a uniform one. So the higher the entropy, the more uncertain the classifier is. So we rank instances in descending order of entropy and pick the top 5 to request human labels. Stopping *UncertaintyAL* cannot be naturally done as with *co-testing*. A less appealing solution is to set a threshold based on the uncertainty measure.

RandomAL. This is a random selection based active leaning baseline. 5 instances are selected randomly during a single iteration. There is no obvious way to stop *RandomAL* although one can use a fixed number of instances as a threshold, a number that might be related to the budget of a project.

The next three baselines aim to investigate the benefits of incorporating features from the global view into the local classifier. They are inspired by recent advances in using cluster level features to compensate for the sparseness of lexical features [11, 14]. Specifically, we use the distributional similarity as a distance measure to build a phrase hierarchy using *Complete Linkage*. The threshold for cutting the hierarchy into clusters is determined by its ability to place the initial seeds into a single cluster. We refer the reader to [13] for more details about how a threshold is selected. If a phrase is a member of cluster *c*, we will assign a cluster feature *phraseCluster=c*.

UncertaintyAL+. The only difference between *UncertaintyAL+* and *UncertaintyAL* is its incorporation of cluster features in building its classifier. This is essentially the active learning approach presented by [11].

SPCo-Testing+. It differs from *SPCo-Testing* only in its sequence view classifier, which is trained with additional phrase cluster features.

LGCo-Testing+. It differs from *LGCo-Testing* only in its local view classifier, which is trained with additional phrase cluster features.

7. Experiments

7.1 Experimental Setting

We experiment with the *nwire* and *bnews* genres of the ACE 2004 data set, which are benchmark evaluation data for relation extraction. There are 4374 relation instances and about 45K *non-relation* instances. Documents are preprocessed with the Stanford parser [4] and chunklink [5] to facilitate feature extraction. Note that following most previous work, we use the hand labeled entities in all our experiments.

We do 5-fold cross-validation as most previous supervised systems do. Each round of cross-validation is repeated 10 times with randomly selected seeds. So, a total of 50 runs are performed (5 subsets times 10 experiments). We report average results of these 50 runs. Note that we do not experiment with the DISC (discourse) relation type which is syntactically different from other relations and was removed from ACE after 2004.

The size of unlabeled data is approximately 36K instances ($45K \div 5 \times 4$). Each iteration selects 5 instances to request human labels and 200 iterations are performed. So a total of 1,000 instances are presented to our annotator. This setting simulates satisfying a customer's demand for an adaptive relation extractor in a few hours. Assuming two entities and their contexts (the annotation unit) are highlighted and an annotator only needs to mark it as a *target relation* or not, 4 instances per minute should be a reasonable annotation speed. And assuming that our annotator takes a 10-minute break in every hour, he or she can annotate 200 instances per hour. We are now ready to test the feasibility and quality of relation type extension in a few hours.

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

[5] http://ilk.uvt.nl/team/sabine/chunklink/README.html

7.2 Results

We evaluate active learning on the *target relation.* Penalty will be given to cases where we predict *target* as *auxiliary* or *non-relation,* and vice versa. To measure the reduction of annotation cost, we compare active learning with the results of [14], which is a state-of-the-art feature-based supervised system. We use the number of labeled instances to approximate the cost of active learning. So we list in Table 5 the F1 difference between an active learning system with different number of labeled instances and the supervised system trained on the entire corpus.

The results of *LGCo-Testing* are simply based on the local classifier's predictions on the test set. For the *SPCo-Testing* system, a third classifier is trained with the full feature set to get test results. There is a large gap between *RandomAL* and the supervised system (40% F1 difference regardless of the number of labeled instances). So we do not include its results in Table 5. The three baselines with cluster level features perform similarly as their corresponding baselines without cluster phrases, e.g. *UncertaintyAL+* and *UncertaintyAL* perform similarly. We only report results for systems without cluster features.

7.3 Analyses

Comparing active learning with supervised learning: *LGCo-Testing* trained with 1,000 labeled examples achieves results comparable to supervised learning trained with more than 35K labeled instances in both the *binary* and the *multi-class* settings. This is true even for the two most frequent relations in ACE 2004, EMP-ORG and PHYS (about 1.6K instances for EMP-ORG and 1.2K for PHYS). This represents a substantial reduction in instances annotated of 97%. So assuming our annotation speed is 200 instances per hour, we can build in five hours a competitive system for EMP-ORG and a slightly weak system for PHYS. Moreover, we can build comparable systems for the other four relations in less than 5 hours. Much of the contribution, as depicted in Fig. 1, can be attributed to the sharp increase in precision during early stages and the steady improvement of recall in later stages of learning.

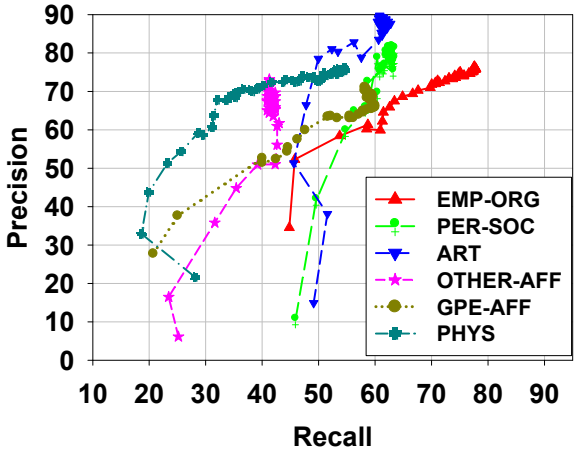

Figure 1: P-R curve of *LGCo-Testing* with the *multi-class* setting. Each dot represents one iteration.

Comparing active learning systems: the clear trend is that *LGCo-Testing* outperforms *UncertaintyAL* and *SPCo-Testing* by a large margin in most cases for both experimental settings. This indicates its superiority in selective sampling for fast system development and adaptation. *SPCo-Testing*, which is based on the feature split at the local level, does not consistently beat the uncertainty based systems. Part of the reason, as we believe, is that the sequence and parsing views are highly correlated. For example, the token sequence feature "*traveled to*" and the dependency path feature "*nsubj' traveled prep_to*" are hardly conditionally independent.

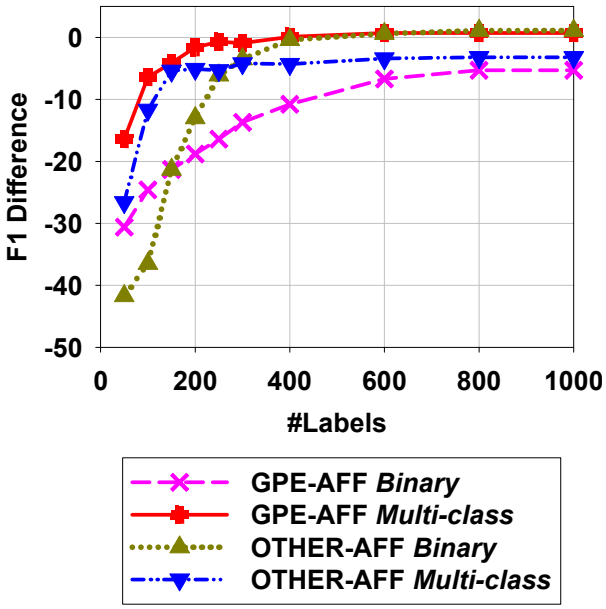

Figure 2. F1 difference of LGCo-Testing for GPE-AFF and OTHER-AFF.

Comparing *LGCo-Testing* in the *multi-class* setting with that in the *binary* setting, we observe that the reduction of annotation cost by incorporating *auxiliary* types is more pronounced in early learning stages (#labels < 200) than in later ones, which is true for most relations. Figure 2 depicts this by plotting the F1 difference (between active learning and supervised learning) of *LGCo-Testing* in the two experimental settings against the number of labels. Besides the two relations GPE-AFF and OTHER-AFF shown in Figure 2, taking ART as a third example relation type, with 50 labels, the *F1 difference* of the *multi-class LGCo-Testing* is *-29.8* while the *binary* one is *-48.4*, which represents a F1 improvement of 19.6 when using *auxiliary* types. As the number of labels increases, the *multi-class* setting incorporates more and more *auxiliary* instances, which might decrease the priors for the *target* relations. Hence the improvement for the *target* relation degrades in later learning stages.

Table 5: F1 difference (in percentage) = *F1 of active learning* minus *F1 of supervised learining*. Bold numbers indicate the best performance. UN := *UncertaintyAL.* SP := *SPCo-Testing.* LG := *LGCo-Testing*

#Labels	EMP-ORG (Supervised = 77.6)						PHYS (Supervised = 66.9)					
	Binary			*Multi-class*			*Binary*			*Multi-class*		
	UN	SP	LG	UN	SP	LG	UN	SP	LG	UN	SP	LG
50	-32.2	-44.8	**-30.7**	-41.2	-29.7	**-24.1**	-42.3	-55.2	**-41.2**	-41.5	-39.0	**-40.3**
100	-22.4	-27.5	**-20.4**	-33.6	-23.4	**-18.8**	-38.7	-51.6	**-36.3**	-35.0	-37.1	**-33.2**
150	-15.0	-24.2	**-14.7**	-27.4	-23.0	**-15.7**	-31.9	-52.2	**-31.2**	-29.8	-33.0	**-26.6**
200	-12.8	-21.7	**-11.4**	-25.0	-18.3	**-12.4**	-27.5	-51.7	**-27.0**	-28.8	-28.9	**-23.5**
250	-11.3	-18.5	**-9.8**	-22.3	-15.7	**-9.1**	-24.0	-51.2	**-22.6**	-25.8	-27.8	**-20.2**
300	-9.8	-18.7	**-8.0**	-20.2	-15.4	**-6.8**	-20.9	-50.3	**-19.8**	-24.6	-24.1	**-19.4**
400	-7.4	-15.9	**-5.4**	-16.6	-12.1	**-5.1**	-16.9	-46.1	**-16.0**	-22.9	-18.1	**-15.7**
600	-4.9	-12.6	**-2.5**	-11.3	-8.8	**-2.7**	**-9.8**	-42.5	-9.9	-17.7	-10.8	**-7.2**
800	-2.2	-8.7	**-1.4**	-8.7	-5.0	**-0.7**	-7.0	-40.0	**-7.0**	-16.9	-5.8	**-4.0**
1000	-1.5	-5.0	**-0.7**	-7.6	-1.5	**-0.7**	-5.9	-38.2	**-4.8**	-14.4	-3.3	**-3.0**

#Labels	GPE-AFF (Supervised = 63.3)						PER-SOC (Supervised = 70.3)					
	Binary			*Multi-class*			*Binary*			*Multi-class*		
	UN	SP	LG	UN	SP	LG	UN	SP	LG	UN	SP	LG
50	-34.9	-48.3	**-30.6**	-48.8	-41.7	**-16.4**	-31.3	-57.5	**-28.6**	-48.0	-37.4	**-26.8**
100	-26.9	-44.0	**-24.6**	-48.2	-33.3	**-6.4**	-4.6	-40.8	**-3.8**	-36.8	-34.1	**-11.3**
150	-24.1	-43.1	**-21.3**	-45.1	-26.8	**-4.1**	-4.6	-25.3	**-2.1**	-32.9	-29.1	**-5.8**
200	-19.2	-40.2	**-18.8**	-40.0	-25.3	**-1.6**	-2.6	-19.2	**+0.5**	-24.5	-23.9	**-2.6**
250	-18.7	-39.7	**-16.4**	-39.0	-22.1	**-0.7**	-1.1	-15.0	**+2.1**	-16.9	-16.8	**-3.7**
300	-15.6	-38.1	**-13.7**	-36.8	-19.6	**-0.9**	-0.8	-11.1	**+2.5**	-15.7	-9.7	**-0.7**
400	**-10.5**	-35.2	-10.8	-32.1	-14.5	**+0.1**	-1.7	-6.7	**+3.0**	-14.6	-4.0	**-0.5**
600	-7.8	-33.3	**-6.7**	-31.1	-9.0	**+0.7**	-1.4	-1.6	**+2.1**	-9.4	0.0	**+ 0.1**
800	-6.9	-30.1	**-5.3**	-28.6	-7.6	**+0.7**	-1.4	0.0	**+2.5**	-5.8	-1.3	**-0.6**
1000	-6.9	-29.6	**-5.3**	-25.0	-7.4	**+0.7**	-1.4	0.0	**+2.0**	-5.1	**0.0**	-0.3

#Labels	ART (Supervised = 73.4)						OTHER-AFF (Supervised = 52.2)					
	Binary			*Multi-class*			*Binary*			*Multi-class*		
	UN	SP	LG	UN	SP	LG	UN	SP	LG	UN	SP	LG
50	-50.9	-66.9	**-48.4**	-48.0	-43.5	**-29.8**	-46.3	-39.7	**-41.7**	-40.0	-34.1	**-26.6**
100	-31.1	-53.5	**-26.9**	-39.9	-36.2	**-13.0**	-43.9	-37.3	**-36.5**	-43.2	-37.5	**-11.7**
150	-17.3	-51.2	**-14.8**	-28.5	-28.3	**-7.1**	-41.9	**-20.5**	-21.3	-40.0	-33.4	**-5.4**
200	-16.7	-48.1	**-8.0**	-23.5	-24.1	**-3.5**	-41.2	**-12.2**	-13.0	-33.4	-29.4	**-5.1**
250	-15.2	-45.7	**-4.0**	-18.1	-22.5	**-2.6**	-34.5	-12.9	**-6.1**	-28.6	-26.5	**-5.3**
300	-13.4	-47.0	**-1.9**	-14.5	-19.9	**-1.9**	-32.3	-12.9	**-3.5**	-27.6	-10.3	**-4.2**
400	-12.1	-43.9	**-3.1**	-13.1	-13.0	**-1.7**	-19.3	-12.9	**-0.4**	-25.4	-5.1	**-4.3**
600	-8.0	-40.4	**-1.2**	-9.1	-5.2	**-0.6**	-8.3	-12.9	**+0.6**	-12.4	**-2.3**	-3.4
800	-4.2	-40.4	**-0.9**	-5.2	-4.8	**-0.6**	-5.9	-12.9	**+1.1**	-7.3	-4.0	**-3.2**
1000	-3.1	-40.4	**-0.9**	-4.1	-7.1	**-0.6**	-5.9	-12.9	**+1.1**	-6.3	-4.1	**-3.2**

As mentioned earlier, the three baselines with cluster level features perform similarly as and do not show obvious advantage over their corresponding baselines without cluster level phrases, which further indicate that stirring global and local views together would not fully utilize the strength of the global view as *co-testing*, at least for the task of relation extraction.

To compare *LGCo-Testing* with semi-supervised learning, we simply take the best results reported in our earlier work [13] as we used pattern clusters based on distributional similarities in bootstrapping, though we evaluated our system only on relations between names and only reported results for EMP-ORG, PHYS and PER-SOC. Our best F1 scores are 60 (EMP-ORG), 37 (PER-SOC), and 28 (PHYS), which are much lower than *LGCo-Testing* with 1000 labels. However, semi-supervised learning systems do not require human labeled examples except for the seeds. It is impressive that with just a few seeds, semi-supervised learning can achieve F1 of 60 for the EMP-ORG relation. Combing it with active learning to further reduce the annotation cost is definitely a promising research avenue of our future work.

8. Related Work

To the best of our knowledge, the literature on the topic of using active learning for relation type extension is rather limited. Our work is first motivated by *co-training* [1] and *co-testing* [12] which provide us with a solid theoretical foundation.

Our global view is mostly triggered by recent advances in using cluster level features for generalized discriminative models, including using word clusters [11] and phrase clusters [10] for name tagging and using word clusters for relation extraction [3, 14].

Our *multi-class* setting is similar to the transfer learning setting of [5], namely building up a system for a *target* relation with a few *target* instances and all *auxiliary* instances. They removed *auxiliary* instances from their evaluation data while we preserved *auxiliary* instances in our evaluation data, which unfortunately hinders a direct and fair comparison between their system and ours.

The current study builds on our earlier work [13] where we used pattern clusters to guide the semi-supervised learning of relation patterns. The notion of pattern similarity is similar, though its application is quite different.

9. Conclusion

We have presented *LGCo-Testing*, a multi-view active learning approach for relation type extension based on *local* and *global* views. Evaluation results showed that *LGCo-Testing* can reduce annotation cost by 97% while maintaining the performance level of supervised learning. It has prepared us well to apply active learning to real world relation type extension tasks. Combining it with semi-supervised learning to further reduce annotation cost is another promising research avenue.

10. ACKNOWLEDGMENTS

This work is supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory (AFRL) contract number FA8650-10-C-7058. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copy-right annotation thereon. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL, or the U.S. Government.

11. REFERENCES

[1] Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*.

[2] Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of HLT/EMNLP*.

[3] Yee Seng Chan and Dan Roth. 2010. Exploiting background knowledge for relation extraction. In *Proc. of COLING*.

[4] Michael Collins and Yoram Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proc. of EMNLP-99*.

[5] Jing Jiang. 2009. Multi-task transfer learning for weakly-supervised relation extraction. In *Proceedings of ACL-IJCNLP-09*.

[6] Jing Jiang and ChengXiang Zhai. 2007. A systematic exploration of the feature space for relation extraction. In *Proceedings of HLT-NAACL-07*.

[7] Rosie Jones , Rayid Ghani , Tom Mitchell , Ellen Riloff. 2003. Active learning for information extraction with multiple view feature sets. In *ECML 2003 Workshop on Adaptive Text Extraction and Mining*.

[8] Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *Proceedings of ACL-04*.

[9] Zornista Kozareva and Eduard Hovy. 2010. Not all seeds are equal: Measuring the quality of text mining seeds. In *NAACL-10*.

[10] Dekang Lin and Xiaoyun Wu. 2009. Phrase Clustering for Discriminative Learning. In *Proc. of ACL-09*.

[11] Scott Miller, Jethran Guinness and Alex Zamanian. 2004. Name Tagging with Word Clusters and Discriminative Training. In *Proc. of HLT-NAACL*.

[12] Ion Muslea, Steve Minton, and Craig Knoblock. 2000. Selective sampling with redundant views. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*.

[13] Ang Sun and Ralph Grishman. 2010. Semi-supervised Semantic Pattern Discovery with Guidance from Unsupervised Pattern Clusters. In *Proc. of COLING-10*.

[14] Ang Sun and Ralph Grishman. 2011. Semi-supervised Relation Extraction with Large-scale Word Clustering. In *Proc. of ACL-11*.

[15] Vishnu Vyas, Patrick Pantel, Eric Crestan. 2009. Helping Editors Choose Better Seed Sets for Entity Set Expansion. In *Proceedings of CIKM-09*.

[16] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.

[17] Min Zhang, Jie Zhang, Jian Su, and GuoDong Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of COLING-ACL-06*.

[18] Zhu Zhang. (2004). Weakly supervised relation classification for information extraction. In *Proc. of CIKM'2004*.

[19] Shubin Zhao and Ralph Grishman. 2005. Extracting relations with integrated information using kernel methods. In *Proceedings of ACL*.

[20] Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of ACL-05*.

[21] 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 EMNLPCoNLL-07*.