Compensating for Annotation Errors in Training a Relation Extractor

Bonan Min
New York University
715 Broadway, 7th floor
New York, NY 10003 USA
min@cs.nyu.edu

Ralph Grishman
New York University
715 Broadway, 7th floor
New York, NY 10003 USA
grishman@cs.nyu.edu

Abstract

The well-studied supervised Relation Extraction algorithms require training data that is accurate and has good coverage. To obtain such a gold standard, the common practice is to do independent double annotation followed by adjudication. This takes significantly more human effort than annotation done by a single annotator. We do a detailed analysis on a snapshot of the ACE 2005 annotation files to understand the differences between single-pass annotation and the more expensive nearly three-pass process, and then propose an algorithm that learns from the much cheaper single-pass annotation and achieves a performance on a par with the extractor trained on multi-pass annotated data. Furthermore, we show that given the same amount of human labor, the better way to do relation annotation is not to annotate with high-cost quality assurance, but to annotate more.

1. Introduction

Relation Extraction aims at detecting and categorizing semantic relations between pairs of entities in text. It is an important NLP task that has many practical applications such as answering factoid questions, building knowledge bases and improving web search.

Supervised methods for relation extraction have been studied extensively since rich annotated linguistic resources, e.g. the Automatic Content Extraction[1] (ACE) training corpus, were released. We will give a summary of related methods in section 2. Those methods rely on accurate and complete annotation. To obtain high quality annotation, the common wisdom is to let two annotators independently annotate a corpus, and then asking a senior annotator to adjudicate the disagreements[2]. This annotation procedure roughly requires 3 passes[3] over the same corpus. Therefore it is very expensive. The ACE 2005 annotation on relations is conducted in this way.

In this paper, we analyzed a snapshot of ACE training data and found that each annotator missed a significant fraction of relation mentions and annotated some spurious ones. We found that it is possible to separate most missing examples from the vast majority of true-negative unlabeled examples, and in contrast, most of the relation mentions that are adjudicated as incorrect contain useful expressions for learning a relation extractor. Based on this observation, we propose an algorithm that purifies negative examples and applies transductive inference to utilize missing examples during the training process on the single-pass annotation. Results show that the extractor trained on single-pass annotation with the proposed algorithm has a performance that is close to an extractor trained on the 3-pass annotation. We further show that the proposed algorithm trained on a single-pass annotation on the complete set of documents has a higher performance than an extractor trained on 3-pass annotation on 90% of the documents in the same corpus, although the effort of doing a single-pass annotation over the entire set costs less than half that of doing 3 passes over 90% of the documents. From the perspective of learning a high-performance relation extractor, it suggests that a better way to do relation annotation is not to annotate with a high-cost quality assurance, but to annotate more.

[1] http://www.itl.nist.gov/iad/mig/tests/ace/

[2] The senior annotator also found some missing examples as shown in figure 1.

[3] In this paper, we will assume that the adjudication pass has a similar cost compared to each of the two first-passes. The adjudicator may not have to look at as many sentences as an annotator, but he is required to review all instances found by both annotators. Moreover, he has to be more skilled and may have to spend more time on each instance to be able to resolve disagreements.

Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pages 194–203,
Avignon, France, April 23 - 27 2012.

2. Background

2.1 Supervised Relation Extraction

One of the most studied relation extraction tasks is the ACE relation extraction evaluation sponsored by the U.S. government. ACE 2005 defined 7 major entity types, such as PER (Person), LOC (Location), ORG (Organization). A relation in ACE is defined as an ordered pair of entities appearing in the same sentence which expresses one of the predefined relations. ACE 2005 defines 7 major relation types and more than 20 subtypes. Following previous work, we ignore sub-types in this paper and only evaluate on types when reporting relation classification performance. Types include General-affiliation (GEN-AFF), Part-whole (PART-WHOLE), Person-social (PER-SOC), etc. ACE provides a large corpus which is manually annotated with entities (with coreference chains between entity mentions annotated), relations, events and values. Each mention of a relation is tagged with a pair of entity mentions appearing in the same sentence as its arguments. More details about the ACE evaluation are on the ACE official website.

Given a sentence *s* and two entity mentions arg_1 and arg_2 contained in *s*, a candidate relation mention *r* with argument arg_1 preceding arg_2 is defined as $r=(s, arg_1, arg_2)$. The goal of Relation Detection and Classification (RDC) is to determine whether *r* expresses one of the types defined. If so, classify it into one of the types. Supervised learning treats RDC as a classification problem and solves it with supervised Machine Learning algorithms such as MaxEnt and SVM. There are two commonly used learning strategies (Sun et al., 2011). Given an annotated corpus, one could apply a **flat** learning strategy, which trains a single multi-class classifier on training examples labeled as one of the relation types or *not-a-relation*, and apply it to determine its type or output *not-a relation* for each candidate relation mention during testing. The examples of each type are the relation mentions that are tagged as instances of that type, and the *not-a-relation* examples are constructed from pairs of entities that appear in the same sentence but are not tagged as any of the types. Alternatively, one could apply a **hierarchical** learning strategy, which trains two classifiers, a binary classifier *RD* for relation detection and the other a multi-class classifier *RC* for relation classification. *RD* is trained by grouping tagged relation mentions of all types as positive instances and using all the *not-a-relation* cases (same as described above) as negative examples. *RC* is trained on the annotated examples with their tagged types. During testing, *RD* is applied first to identify whether an example expresses some relation, then RC is applied to determine the most likely type only if it is detected as correct by *RD*.

State-of-the-art supervised methods for relation extraction also differ from each other on data representation. Given a relation mention, feature-based methods (Miller et al., 2000; Kambhatla, 2004; Boschee et al., 2005; Grishman et al., 2005; Zhou et al., 2005; Jiang and Zhai, 2007; Sun et al., 2011) extract a rich list of structural, lexical, syntactic and semantic features to represent it; in contrast, the kernel based methods (Zelenko et al., 2003; Bunescu and Mooney, 2005a; Bunescu and Mooney, 2005b; Zhao and Grishman, 2005; Zhang et al., 2006a; Zhang et al., 2006b; Zhou et al., 2007; Qian et al., 2008) represent each instance with an object such as augmented token sequences or a parse tree, and used a carefully designed kernel function, e.g. subsequence kernel (Bunescu and Mooney, 2005b) or convolution tree kernel (Collins and Duffy, 2001), to calculate their similarity. These objects are usually augmented with features such as semantic features.

In this paper, we use the hierarchical learning strategy since it simplifies the problem by letting us focus on relation detection only. The relation classification stage remains unchanged and we will show that it benefits from improved detection. For experiments on both relation detection and relation classification, we use SVM[4] (Vapnik 1998) as the learning algorithm since it can be extended to support transductive inference as discussed in section 4.3. However, for the analysis in section 3.2 and the purification preprocess steps in section 4.2, we use a MaxEnt[5] model since it outputs probabilities[6] for its predictions. For the choice of features, we use the full set of features from Zhou et al. (2005) since it is reported to have a state-of-the-art performance (Sun et al., 2011).

2.2 ACE 2005 annotation

The ACE 2005 training data contains 599 articles

[4] SVM-Light is used. http://svmlight.joachims.org/
[5] OpenNLP MaxEnt package is used. http://maxent.sourceforge.net/about.html
[6] SVM also outputs a value associated with each prediction. However, this value cannot be interpreted as probability.

from newswire, broadcast news, weblogs, usenet newsgroups/discussion forum, conversational telephone speech and broadcast conversations. The annotation process is conducted as follows: two annotators working independently annotate each article and complete all annotation tasks (entities, values, relations and events). After two annotators both finished annotating a file, all discrepancies are then adjudicated by a senior annotator. This results in a high-quality annotation file. More details can be found in the documentation of ACE 2005 Multilingual Training Data V3.0.

Since the final release of the ACE training corpus only contains the final adjudicated annotations, in which all the traces of the two first-pass annotations are removed, we use a snapshot of almost-finished annotation, ACE 2005 Multilingual Training Data V3.0, for our analysis. In the remainder of this paper, we will call the two independent first-passes of annotation *fp1* and *fp2*. The higher-quality data done by merging *fp1* and *fp2* and then having disagreements adjudicated by the senior annotator is called *adj*. From this corpus, we removed the files that have not been completed for all three passes. On the final corpus consisting of 511 files, we can differentiate the annotations on which the three annotators have agreed and disagreed.

A notable fact of ACE relation annotation is that it is done with arguments from the list of annotated entity mentions. For example, in a relation mention *tyco's ceo and president dennis kozlowski* which expresses an *EMP-ORG* relation, the two arguments *tyco* and *dennis kozlowski* must have been tagged as entity mentions previously by the annotator. Since *fp1* and *fp2* are done on all tasks independently, their disagreement on entity annotation will be propagated to relation annotation; thus we need to deal with these cases specifically.

3. Analysis of data annotation

3.1 General statistics

As discussed in section 2, relation mentions are annotated with entity mentions as arguments, and the lists of annotated entity mentions vary in *fp1*, *fp2* and *adj*. To estimate the impact propagated from entity annotation, we first calculate the ratio of overlapping entity mentions between entities annotated in *fp1/fp2* with *adj*. We found that *fp1/fp2* each agrees with *adj* on around 89% of the entity mentions. Following up, we checked the relation mentions[7] from *fp1* and *fp2* against the adjudicated list of entity mentions from *adj* and found that 682 and 665 relation mentions respectively have at least one argument which doesn't appear in the list of adjudicated entity mentions.

Given the list of relation mentions with both arguments appearing in the list of adjudicated entity mentions, figure 1 shows the inter-annotator agreement of the ACE 2005 relation annotation. In this figure, the three circles represent the list of relation mentions in *fp1*, *fp2* and *adj*, respectively.

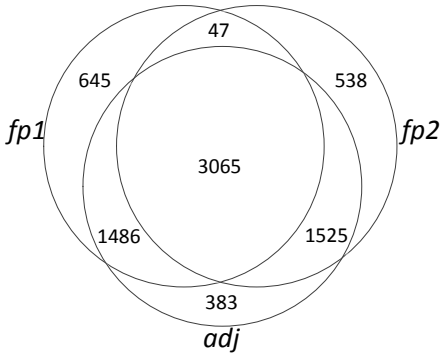

Figure 1. Inter-annotator agreement of ACE 2005 relation annotation. Numbers are the distinct relation mentions whose both arguments are in the list of adjudicated entity mentions.

It shows that each annotator missed a significant number of relation mentions annotated by the other. Considering that we removed 682/665 relation mentions from *fp1/fp2* because we generate this figure based on the list of adjudicated entity mentions, we estimate that *fp1* and *fp2* both missed around 18.3-28.5%[8] of the relation mentions. This clearly shows that both of the annotators missed a significant fraction of the relation mentions. They also annotated some spurious relation mentions (as adjudicated in *adj*), although the fraction is smaller (close to 10% of all relation mentions in *adj*).

ACE 2005 relation annotation guidelines (ACE English Annotation Guidelines for Relations, version 5.8.3) defined 7 syntactic classes and the *other* class. We plot the distribution of syntactic classes of the annotated

[7] This is done by selecting the relation mentions whose both arguments are in the list of adjudicated entity mentions.

[8] We calculate the lower bound by assuming that the 682 relation mentions removed from *fp1* are found in *fp2*, although with different argument boundary and headword tagged. The upper bound is calculated by assuming that they are all irrelevant and erroneous relation mentions.

relations in figure 2 (3 of the classes, accounting together for less than 10% of the cases, are omitted) and the *other* class. It seems that it is generally easier for the annotators to find and agree on relation mentions of the type *Preposition/PreMod/Possessives* but harder to find and agree on the ones belonging to *Verbal* and *Other*. The definition and examples of these syntactic classes can be found in the annotation guidelines.

In the following sections, we will show the analysis on *fp1* and *adj* since the result is similar for *fp2*.

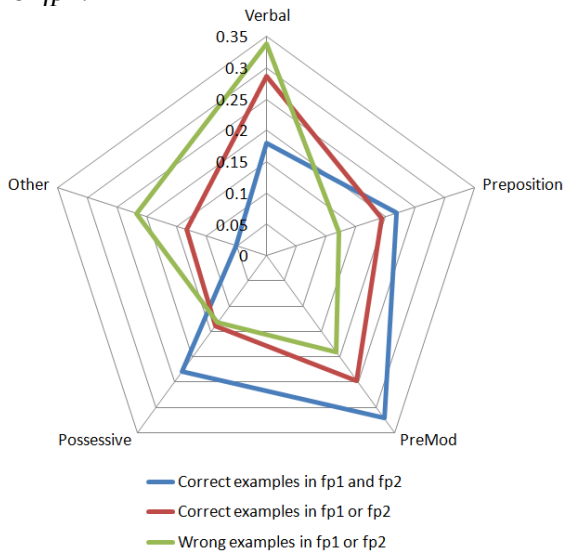

Figure 2. Percentage of examples of major syntactic classes.

3.2 Why the differences?

To understand what causes the missing annotations and the spurious ones, we need methods to find how similar/different the false positives are to true positives and also how similar/different the false negatives (missing annotations) are to true negatives. If we adopt a good similarity metric, which captures the structural, lexical and semantic similarity between relation mentions, this analysis will help us to understand the similarity/difference from an extraction perspective.

We use a state-of-the-art feature space (Zhou et al., 2005) to represent examples (including all correct examples, erroneous ones and untagged examples) and use MaxEnt as the weight learning model since it shows competitive performance in relation extraction (Jiang and Zhai, 2007) and outputs probabilities associated with each prediction. We train a MaxEnt model for relation detection on true positives and true negatives, which respectively are the subset of correct examples annotated by *fp1* (and adjudicated as correct ones) and negative examples that are not annotated in *adj*, and use it to make predictions on the mixed pool of correct examples, missing examples and spurious ones.

To illustrate how distinguishable the missing examples (false negatives) are from the true negative ones, 1) we apply the MaxEnt model on both false negatives and true negatives, 2) put them together and rank them by the model-predicted probabilities of being positive, 3) calculate their relative rank in this pool. We plot the Cumulative distribution of frequency (CDF) of the ranks (as percentages in the mixed pools) of false negatives in figure 3. We took similar steps for the spurious ones (false positives) and plot them in figure 3 as well (However, they are ranked by model-predicted probabilities of being negative).

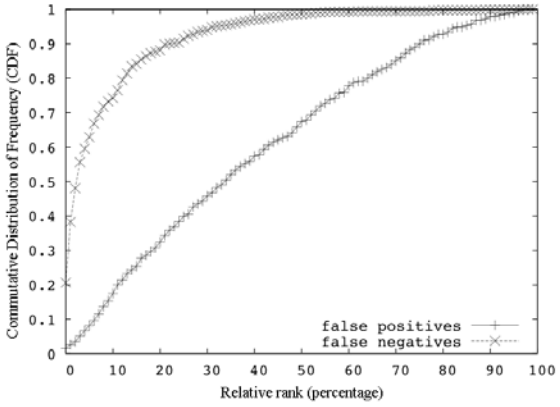

Figure 3: cumulative distribution of frequency (CDF) of the relative ranking of model-predicted probability of being positive for false negatives in a pool mixed of false negatives and true negatives; and the CDF of the relative ranking of model-predicted probability of being negative for false positives in a pool mixed of false positives and true positives.

For false negatives, it shows a highly skewed distribution in which around 75% of the false negatives are ranked within the top 10%. That means the missing examples are lexically, structurally or semantically similar to correct examples, and are distinguishable from the true negative examples. However, the distribution of false positives (spurious examples) is close to uniform (flat curve), which means they are generally indistinguishable from the correct examples.

3.3 Categorize annotation errors

The automatic method shows that the errors (spurious annotations) are very similar to the correct examples but provides little clue as to why that is the case. To understand their causes, we sampled 65 examples from *fp1* (10% of the 645 errors), read the sentences containing these

Category	Percentage	Example		
		Relation Type	Sampled text of spurious examples in *fp1*	Notes (examples are similar ones in *adj* for comparison)
Duplicate relation mention for coreferential entity mentions	49.2%	ORG-AFF	*… his budding friendship with US President George W. Bush in the face of …*	*… his budding friendship with US President George W. Bush in the face of …*
Correct	20%	PHYS	*Hundreds of thousands of demonstrators took to the streets in Britain…*	
		PER-SOC	*The dead included the quack doctor, 55-year-old Nityalila Naotia, his teenaged son and…*	(Symmetric relation) *The dead included the quack doctor, 55-year-old Nityalila Naotia, his teenaged son*
Argument not in list	15.4%	PER-SOC	*Putin had even secretly invited British Prime Minister Tony Blair, Bush's staunchest backer in the war on Iraq…*	
Violate reasonable reader rule	6.2%	PHYS	*"The amazing thing is they are going to turn San Francisco into ground zero for every criminal who wants to profit at their chosen profession", Paredes said.*	
Errors	6.1%	PART-WHOLE	*…a likely candidate to run Vivendi Universal's entertainment unit in the United States…*	Arguments are tagged reversed
		PART-WHOLE	*Khakamada argued that the United States would also need Russia's help "to make the new Iraqi government seem legitimate.*	Relation type error
illegal promotion through "blocked" categories	3%	PHYS	*Up to 20,000 protesters thronged the plazas and streets of San Francisco, where…*	*Up to 20,000 protesters thronged the plazas and streets of San Francisco, where…*

Table 1. Categories of spurious relation mentions in *fp1 (on a sample of 10%* of relation mentions), ranked by the percentage of the examples in each category. In the sample text, red text (also marked with dotted underlines) shows head words of the first arguments and the underlined text shows head words of the second arguments.

erroneous relation mentions and compared them to the correct relation mentions in the same sentence; we categorized these examples and show them in table 1. The most common type of error is *duplicate relation mention for coreferential entity mentions*. The first row in table 1 shows an example, in which there is a relation ORG-AFF tagged between *US* and *George W. Bush* in *adj*. Because *President* and *George W. Bush* are coreferential, the example <*US, President* > from *fp1* is adjudicated as incorrect. This shows that if a relation is expressed repeatedly across relation mentions whose arguments are coreferential, the adjudicator only tags one of the relation mentions as correct, although the other is correct too. This shared the same principle with another type of error *illegal promotion through "blocked" categories* [9] as defined in the annotation guideline. The second largest category is *correct*, by which we mean the example is a correct relation mention and the adjudicator made a mistake. The third largest category is *argument not in list*, by which we mean that at least one of the arguments is not in the list of adjudicated entity mentions.

Based on Table 1, we can see that as many as 72%-88% of the examples which are adjudicated as incorrect are actually correct if viewed from a relation learning perspective, since most of them contain informative expressions for tagging relations. The annotation guideline is designed to ensure high quality while not imposing too much burden on human annotators. To reduce annotation effort, it defined rules such as *illegal promotion through "blocked" categories*. The annotators' practice suggests that they are following another rule not to annotate *duplicate relation mention for coreferential entity mentions*. This follows the similar principle of reducing annotation effort but is not explicitly stated in the guideline: to avoid propagation of a relation through a coreference chain. However, these examples are useful for learning more ways to express a relation. Moreover, even for the erroneous examples (as shown in table 1 as *violate reasonable reader rule* and *errors*), most of them have some level of similar structures or semantics to the targeted relation. Therefore, it is very hard to distinguish them without human proofreading.

[9] For example, in sentence *Smith went to a hotel in Brazil*, *(Smith, hotel)* is a taggable PHYS Relation but *(Smith, Brazil)* is not, because to get the second relationship, one would have to "promote" *Brazil* through *hotel*. For the precise definition of annotation rules, please refer to ACE (Automatic Content Extraction) English Annotation Guidelines for Relations, version 5.8.3.

Exp #	Training data	Testing data	Detection (%)			Classification (%)		
			Precision	Recall	F1	Precision	Recall	F1
1	*fp1*	*adj*	83.4	60.4	70.0	75.7	54.8	63.6
2	*fp2*	*adj*	83.5	60.5	70.2	76.0	55.1	63.9
3	*adj*	*adj*	80.4	69.7	74.6	73.4	63.6	68.2

Table 2. Performance of RDC trained on *fp1/fp2/adj*, and tested on *adj*.

3.4 Why missing annotations and how many examples are missing?

For the large number of missing annotations, there are a couple of possible reasons. One reason is that it is generally easier for a human annotator to annotate correctly given a well-defined guideline, but it is hard to ensure completeness, especially for a task like relation extraction. Furthermore, the ACE 2005 annotation guideline defines more than 20 relation subtypes. These many subtypes make it hard for an annotator to keep all of them in mind while doing the annotation, and thus it is inevitable that some examples are missed.

Here we proceed to approximate the number of missing examples given limited knowledge. Let each annotator annotate n examples and assume that each pair of annotators agrees on a certain fraction p of the examples. Assuming the examples are equally likely to be found by an annotator, therefore the total number of unique examples found by k annotators is $\sum_{i=0}^{k}(1-p)^i n$. If we had an infinite number of annotators ($k \to \infty$), the total number of unique examples will be $\frac{n}{p}$, which is the upper bound of the total number of examples. In the case of the ACE 2005 relation mention annotation, since the two annotators annotate around 4500 examples and they agree on 2/3 of them, the total number of all positive examples is around 6750. This is close to the number of relation mentions in the adjudicated list: 6459. Here we assume the adjudicator is doing a more complex task than an annotator, resolving the disagreements and completing the annotation (as shown in figure 1).

The assumption of the calculation is a little crude but reasonable given the limited number of passes of annotation we have. Recent research (Ji et al, 2010) shows that, by adding annotators for IE tasks, the merged annotation tends to converge after having 5 annotators. To understand the annotation behavior better, in particular whether annotation will converge after adding a few annotators, more passes of annotation need to be collected. We leave this as future work.

4. Relation extraction with low-cost annotation

4.1 Baseline algorithm

To see whether a single-pass annotation is useful for relation detection and classification, we did 5-fold cross validation (5-fold CV) with each of *fp1*, *fp2* and *adj* as the training set, and tested on *adj*. The experiments are done with the same 511 documents we used for the analysis. As shown in table 2, we did 5-fold CV on *adj* for experiment 3. For fairness, we use settings similar to 5-fold CV for experiment 1 and 2. Take experiment 1 as an example: we split both of *fp1* and *adj* into 5 folds, use 4 folds from *fp1* as training data, and 1 fold from *adj* as testing data and does one train-test cycle. We rotate the folds (both training and testing) and repeat 5 times. The final results are averaged over the 5 runs. Experiment 2 was conducted similarly. In the reminder of the paper, 5-fold CV experiments are all conducted in this way.

Table 2 shows that a relation tagger trained on the single-pass annotated data *fp1* performs worse than the one trained on merged and adjudicated data *adj*, with 4.6 points lower F measure in relation detection, and 4.6 points lower relation classification. For detection, precision on *fp1* is 3 points higher than on *adj* but recall is much lower (close to 10 points). The recall difference shows that the missing annotations contain expressions that can help to find more correct examples during testing. The small precision difference indirectly shows that the spurious ones in *fp1* (as adjudicated) do not hurt precision. Performance on classification shows a similar trend because the relation classifier takes the examples predicted by the detector as correct as its input. Therefore, if there is an error, it gets propagated to this stage. Table 2 also shows similar performance differences between *fp2* and *adj*.

In the remainder of this paper, we will discuss a few algorithms to improve a relation tagger trained on single-pass annotated data[10]. Since we

[10] We only use *fp1* and *adj* in the following experiments because we observed that *fp1* and *fp2* are similar in general in the analysis, though a fraction of the annotation in *fp1*

already showed that most of the spurious annotations are not actually errors from an extraction perspective and table 2 shows that they do not hurt precision, we will only focus on utilizing the missing examples, in other words, training with an incomplete annotation.

4.2 Purify the set of negative examples

As discussed in section 2, traditional supervised methods find all pairs of entity mentions that appear within a sentence, and then use the pairs that are not annotated as relation mentions as the negative examples for the purpose of training a relation detector. It relies on the assumption that the annotators annotated all relation mentions and missed no (or very few) examples. However, this is not true for training on a single-pass annotation, in which a significant portion of relation mentions are left not annotated. If this scheme is applied, all of the correct pairs which the annotators missed belong to this "negative" category. Therefore, we need a way to purify the "negative" set of examples obtained by this conventional approach.

Li and Liu (2003) focuses on classifying documents with only positive examples. Their algorithm initially sets all unlabeled data to be negative and trains a Rocchio classifier, selects negative examples which are closer to the negative centroid than positive centroid as the purified negative examples, and then retrains the model. Their algorithm performs well for text classification. It is based on the assumption that there are fewer unannotated positive examples than negative ones in the unlabeled set, so true negative examples still dominate the set of noisy "negative" examples in the purification step.

Based on the same assumption, our purification process consists of the following steps:

1) Use annotated relation mentions as positive examples; construct all possible relation mentions that are not annotated, and initially set them to be negative. We call this noisy data set *D*.
2) Train a MaxEnt relation detection model M_{det} on *D*.
3) Apply M_{det} on all unannotated examples, and rank them by the model-predicted probabilities of being positive,
4) Remove the top *N* examples from *D*.

These preprocessing steps result in a purified data set D_{pure}. We can use D_{pure} for the normal training process of a supervised relation extraction algorithm.

The algorithm is similar to Li and Liu 2003. However, we drop a few noisy examples instead of choosing a small purified subset since we have relatively few false negatives compared to the entire set of unannotated examples. Moreover, after step 3, most false negatives are clustered within the small region of top ranked examples which has a high model-predicated probability of being positive. The intuition is similar to what we observed from figure 3 for false negatives since we also observed very similar distribution using the model trained with noisy data. Therefore, we can purify negatives by removing examples in this noisy subset.

However, the false negatives are still mixed with true negatives. For example, still slightly more than half of the top 2000 examples are true negatives. Thus we cannot simply flip their labels and use them as positive examples. In the following section, we will use them in the form of unlabeled examples to help train a better model.

4.3 Transductive inference on unlabeled examples

Transductive SVM (Vapnik, 1998; Joachims, 1999) is a semi-supervised learning method which learns a model from a data set consisting of both labeled and unlabeled examples. Compared to its popular antecedent SVM, it also learns a maximum margin classification hyperplane, but additionally forces it to separate a set of unlabeled data with large margin. The optimization function of Transductive SVM (TSVM) is the following:

$$
\begin{aligned}
&Minimize\ over\ (y_1^*, ..., y_n^*, \vec{w}, b, \xi_1, ..., \xi_n, \xi_1^*, ..., \xi_k^*):\\
&\qquad \frac{1}{2}||\vec{w}||^2 + C\sum_{i=0}^{n}\xi_i + C^*\sum_{j=0}^{k}\xi_j^*\\
&subject\ to: \quad \forall_{i=1}^{n} : y_i[\vec{w}\cdot\vec{x}_i + b] \geq 1 - \xi_i\\
&\qquad \forall_{j=1}^{k} : y_j^*[\vec{w}\cdot\vec{x}_j^* + b] \geq 1 - \xi_j^*\\
&\qquad \forall_{i=1}^{n} : \xi_i > 0\\
&\qquad \forall_{j=1}^{k} : \xi_j^* > 0
\end{aligned}
$$

Figure 4. TSVM optimization function for non-separable case (Joachims, 1999)

TSVM can leverage an unlabeled set of examples to improve supervised learning. As shown in section 3, a significant number of relation mentions are missing from the single-pass annotation data. Although it is not possible to find all missing annotations without human effort, we can improve the model by further

and *fp2* is different. Moreover, algorithms trained on them show similar performance.

utilizing the fact that some unannotated examples should have been annotated.

The purification process discussed in the previous section removes *N* examples which have a high density of false negatives. We further utilize the *N* examples as follows:

1) Construct a training corpus D_{hybrid} from D_{pure} by taking a random sample[11] of *N*(1-p)/p* (*p* is the ratio of annotated examples to all examples; *p=0.05* in *fp1*) negatively labeled examples in D_{pure} and setting them to be unlabeled. In addition, the *N* examples removed by the purification process are added back as unlabeled examples.
2) Train TSVM on D_{hybrid}.

The second step trained a model which replaced the detection model in the hierarchical detection-classification learning scheme we used. We will show in the next section that this improves the model.

5. Experiments

Experiments were conducted over the same set of documents on which we did analysis: the 511 documents which have completed annotation in all of the *fp1*, *fp2* and *adj* from the ACE 2005 Multilingual Training Data V3.0. To reemphasize, we apply the hierarchical learning scheme and we focus on improving relation detection while keeping relation classification unchanged (results show that its performance is improved because of the improved detection). We use SVM as our learning algorithm with the full feature set from Zhou et al. (2005).

Baseline algorithm: The relation detector is unchanged. We follow the common practice, which is to use annotated examples as positive ones and all possible untagged relation mentions as negative ones. We sub-sampled the negative data by ½ since that shows better performance.

+purify: This algorithm adds an additional purification preprocessing step (section 4.2) before the hierarchical learning RDC algorithm. After purification, the RDC algorithm is trained on the positive examples and purified negative examples. We set *N=2000*[12] in all experiments.

+tSVM: First, the same purification process of **+purify** is applied. Then we follow the steps described in section 4.3 to construct the set of unlabeled examples, and set all the rest of purified negative examples to be negative. Finally, we train TSVM on both labeled and unlabeled data and replace the relation detection in the RDC algorithm. The relation classification is unchanged.

Table 3 shows the results. All experiments are done with 5-fold cross validation[13] using testing data from *adj*. The first three rows show experiments trained on *fp1*, and the last row (**ADJ**) shows the unmodified RDC algorithm trained on *adj* for comparison. The purification of negative examples shows significant performance gain, 3.7% F1 on relation detection and 3.4% on relation classification. The precision decreases but recall increases substantially since the missing examples are not treated as negatives. Experiment shows that the purification process removes more than 60% of the false negatives. Transductive SVM further improved performance by a relatively small margin. This shows that the latent positive examples can help refine the model. Results also show that transductive inference can find around 17% of missing relation mentions. We notice that the performance of relation classification is improved since by improving relation detection, some examples that do not express a relation are removed. The classification performance on single-pass annotation is close to the one trained on *adj* due to the help from a better relation detector trained with our algorithm.

We also did 5-fold cross validation with a model trained on a fraction of the 4/5 (4 folds) of *adj* data (each experiment shown in table 4 uses 4 folds of *adj* documents for training since one fold is left for cross validation). The documents are sampled randomly. Table 4 shows results for varying training data size. Compared to the results shown in the "+tSVM" row of table 3, we can see that our best model trained on single-pass annotation outperforms SVM trained on 90% of the dual-pass, adjudicated data in both relation detection and classification, although it costs less than half the 3-pass annotation. This suggests that given the same amount of human effort for

[11] We included this large random sample so that the balance of positive to negative examples in the unlabeled set would be similar to that of the labeled data. The test data is not included in the unlabeled set.

[12] We choose 2000 because it is close to the number of relations missed from each single-pass annotation. In practice, it contains more than 70% of the false negatives, and it is less than 10% of the unannotated examples. To estimate how many examples are missing (section 3.4), one should perform multiple passes of independent annotation on a small dataset and measure inter-annotator agreements.

[13] Details about the settings for 5-fold cross validation are in section 4.1.

Algorithm	Detection (%)			Classification (%)		
	Precision	Recall	F1	Precision	Recall	F1
Baseline	83.4	60.4	70.0	75.7	54.8	63.6
+purify	76.8	70.9	73.7	69.8	64.5	67.0
+tSVM	76.4	72.1	**74.2**	69.4	65.2	**67.2**
ADJ (on *adj*)	80.4	69.7	74.6	73.4	63.6	68.2

Table 3. 5-fold cross-validation results. All are trained on *fp1* (except the last row showing the unchanged algorithm trained on *adj* for comparison), and tested on adj. McNemar's test show that the improvement from +purify to +tSVM, and from +tSVM to ADJ are statistically significant (with $p<0.05$).

Percentage of *adj* used	Detection (%)			Classification (%)		
	Precision	Recall	F1	Precision	Recall	F1
60% × 4/5	86.9	41.2	55.8	78.6	37.2	50.5
70% × 4/5	85.5	51.3	64.1	77.7	46.6	58.2
80% × 4/5	83.3	58.1	68.4	75.8	52.9	62.3
90% × 4/5	82.0	64.9	72.5	74.9	59.4	66.2

Table 4. Performance with SVM trained on a fraction of *adj*. It shows 5 fold cross validation results.

relation annotation, annotating more documents with single-pass offers advantages over annotating less data with high quality assurance (dual passes and adjudication).

6. Related work

Dligach et al. (2010) studied WSD annotation from a cost-effectiveness viewpoint. They showed empirically that, with same amount of annotation dollars spent, single-annotation is better than dual-annotation and adjudication. The common practice for quality control of WSD annotation is similar to Relation annotation. However, the task of WSD annotation is very different from relation annotation. WSD requires that every example must be assigned some tag, whereas that is not required for relation tagging. Moreover, relation tagging requires identifying two arguments and correctly categorizing their types.

The purified approach applied in this paper is related to the general framework of learning from positive and unlabeled examples. Li and Liu (2003) initially set all unlabeled data to be negative and train a Rocchio classifier, then select negative examples which are closer to the negative centroid than positive centroid as the purified negative examples. We share a similar assumption with Li and Liu (2003) but we use a different method to select negative examples since the false negative examples show a very skewed distribution, as described in section 5.2.

Transductive SVM was introduced by Vapnik (1998) and later refined in Joachims (1999). A few related methods were studied on the subtask of relation classification (the second stage of the hierarchical learning scheme) in Zhang (2005).

Chan and Roth (2011) observed the similar phenomenon that ACE annotators rarely duplicate a relation link for coreferential mentions. They use an evaluation scheme to avoid being penalized by the relation mentions which are not annotated because of this behavior.

7. Conclusion

We analyzed a snapshot of the ACE 2005 relation annotation and found that each single-pass annotation missed around 18-28% of relation mentions and contains around 10% spurious mentions. A detailed analysis showed that it is possible to find some of the false negatives, and that most spurious cases are actually correct examples from a system builder's perspective. By automatically purifying negative examples and applying transductive inference on suspicious examples, we can train a relation classifier whose performance is comparable to a classifier trained on the dual-annotated and adjudicated data. Furthermore, we show that single-pass annotation is more cost-effective than annotation with high quality assurance.

Acknowledgments

Supported 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 copyright annotation thereon. The views and conclusions contained herein are those of the authors 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.

References

ACE. http://www.itl.nist.gov/iad/mig/tests/ace/

ACE (Automatic Content Extraction) English Annotation Guidelines for Relations, version 5.8.3. 2005. http://projects.ldc.upenn.edu/ace/.

ACE 2005 Multilingual Training Data V3.0. 2005. LDC2005E18. LDC Catalog.

Elizabeth Boschee, Ralph Weischedel, and Alex Zamanian. 2005. Automatic information extraction. In Proceedings of the International Conference on Intelligence Analysis.

Razvan C. Bunescu and Raymond J. Mooney. 2005a. A shortest path dependency kenrel for relation extraction. In Proceedings of HLT/EMNLP-2005.

Razvan C. Bunescu and Raymond J. Mooney. 2005b. Subsequence kernels for relation extraction. In Proceedings of NIPS-2005.

Yee Seng Chan and Dan Roth. 2011. Exploiting Syntactico-Semantic Structures for Relation Extraction. In Proceedings of ACL-2011.

Michael Collins and Nigel Duffy. Convolution Kernels for Natural Language. In Proceedings of NIPS-2001.

Dmitriy Dligach, Rodney D. Nielsen and Martha Palmer. 2010. To annotate more accurately or to annotate more. In Proceedings of Fourth Linguistic Annotation Workshop at ACL 2010

Ralph Grishman, David Westbrook and Adam Meyers. 2005. NYU's English ACE 2005 System Description. In Proceedings of ACE 2005 Evaluation Workshop

Scott Miller, Heidi Fox, Lance Ramshaw, and Ralph Weischedel. 2000. A novel use of statistical parsing to extract information from text In Proceedings of NAACL-2010.

Heng Ji, Ralph Grishman, Hoa Trang Dang and Kira Griffitt. 2010. An Overview of the TAC2010 Knowledge Base Population Track. In Proceedings of TAC-2010

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

Thorsten Joachims. 1999. Transductive Inference for Text Classification using Support Vector Machines. In Proceedings of ICML-1999.

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

Xiao-Li Li and Bing Liu. 2003. Learning to classify text using positive and unlabeled data. In Proceedings of IJCAI-2003.

Longhua Qian, Guodong Zhou, Qiaoming Zhu and Peide Qian. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction . In Proc. of COLING-2008.

Ang Sun, Ralph Grishman and Satoshi Sekine. 2011. Semi-supervised Relation Extraction with Large-scale Word Clustering. In Proceedings of ACL-2011.

Vladimir N. Vapnik. 1998. Statistical Learning Theory. John Wiley.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. Journal of Machine Learning Research.

Min Zhang, Jie Zhang and Jian Su. 2006a. Exploring syntactic features for relation extraction using a convolution tree kernel, In Proceedings of HLT-NAACL-2006.

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

Zhu Zhang. 2005. Mining Inter-Entity Semantic Relations Using Improved Transductive Learning. In Proceedings of ICJNLP-2005.

Shubin Zhao and Ralph Grishman, 2005. Extracting Relations with Integrated Information Using Kern el Methods. In Proceedings of ACL-2005.

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

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 EMNLP/CoNLL-2007.