Partially Supervised Coreference Resolution for Opinion Summarization through Structured Rule Learning

Veselin Stoyanov and **Claire Cardie**
Department of Computer Science
Cornell University
Ithaca, NY 14850, USA
{ves,cardie}@cs.cornell.edu

Abstract

Combining fine-grained opinion information to produce opinion summaries is important for sentiment analysis applications. Toward that end, we tackle the problem of source coreference resolution – linking together source mentions that refer to the same entity. The partially supervised nature of the problem leads us to define and approach it as the novel problem of partially supervised clustering. We propose and evaluate a new algorithm for the task of source coreference resolution that outperforms competitive baselines.

1 Introduction

Sentiment analysis is concerned with extracting attitudes, opinions, evaluations, and sentiment from text. Work in this area has been motivated by the desire to provide information analysis applications in the arenas of government, business, and politics (e.g. Coglianese (2004)). Additionally, sentiment analysis can augment existing NLP applications such as question answering, information retrieval, summarization, and clustering by providing information about sentiment (e.g. Stoyanov et al. (2005), Riloff et al. (2005)). To date, research in the area (see Related Work section) has focused on the problem of extracting sentiment both at the document level (*coarse-grained sentiment information*), and at the level of sentences, clauses, or individual expressions (*fine-grained sentiment information*).

In contrast, our work concerns the *summarization* of fine-grained information about *opinions*. In particular, while recent research efforts have shown that fine-grained opinions (e.g. Riloff and Wiebe (2003), Bethard et al. (2004), Wiebe and Riloff (2005)) as well as their sources (e.g. Bethard et al. (2004), Choi et al. (2005), Kim and Hovy (2005)) can be extracted automatically, little has been done to create *opinion summaries*, where opinions from the same source/target are combined, statistics are computed for each source/target and multiple opinions from the same source to the same target are aggregated. A simple opinion summary is shown in figure 1.[1] We expect that this type of opinion summary, based on fine-grained opinion information, will be important for information analysis applications in any domain where the analysis of opinions is critical.

This paper addresses the problem of opinion summarization by considering the creation of simple opinion summaries like those of figure 1. We propose *source coreference resolution* — the task of determining which mentions of opinion sources refer to the same entity — as the primary mechanism for identifying the set of opinions attributed to each real-world source. For this type of summary, source coreference resolution constitutes an integral step in the process of generating full opinion summaries. For example, given the opinion expressions of figure 1, their polarity, and the associated opinion sources and targets, the bulk of the resulting summary can be produced by recognizing that source mentions "Zacarias Moussaoui", "he", "my", and "Mr. Moussaoui" all refer to the same person; and that source mentions "Mr. Zerkin" and "Zerkin" refer to the same person.[2]

[1] For simplicity, the example summary does not contain any source/target statistics.

[2] In addition, the summary would require the closely related task of target coreference resolution and a means for aggregating the conflicting opinions from *Zerkin* toward *Moussaoui*.

Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pages 336–344,
Sydney, July 2006.

At first glance, source coreference resolution appears equivalent to the task of noun phrase coreference resolution and therefore amenable to traditional coreference resolution techniques (e.g. Ng and Cardie (2002), Morton (2000)). We hypothesize in Section 3, however, that the task is likely to succumb to a better solution by treating it in the context of a new machine learning setting that we refer to as *partially supervised clustering*. In particular, due to high coreference annotation costs, data sets that are annotated with opinion information (like ours) do not typically include supervisory coreference information for *all* noun phrases in a document (as would be required for the application of traditional coreference resolution techniques), but only for noun phrases that act as opinion sources (or targets).

As a result, we define the task of *partially supervised clustering*, the goal of which is to learn a clustering function from a set of partially specified clustering examples (Section 4). We are not aware of prior work on the problem of partially supervised clustering and argue that it differs substantially from that of semi-supervised clustering. We propose an algorithm for partially supervised clustering that extends a rule learner with structure information and is generally applicable to problems that fit the partially supervised clustering definition (Section 5). We apply the algorithm to the source coreference resolution task and evaluate its performance on a standard sentiment analysis data set that includes source coreference chains (Section 6). We find that our algorithm outperforms highly competitive baselines by a considerable margin – B^3 score of 83.2 vs. 81.8 and 67.1 vs. 60.9 F1 score for the identification of positive source coreference links.

[Source Zacarias Moussaoui] [– *complained*] at length today about [Target his own lawyer], telling a federal court jury that [Target he] was [– *more interested in achieving fame than saving Moussaoui's life*].

Mr. Moussaoui said he was appearing on the witness stand to tell the truth. And one part of the truth, [Source he] said, is that [Target sending him to prison for life] would be "[– *a greater punishment*] than being sentenced to death."

"[– [Target *You*] *have put your interest ahead of* [Source *my*] *life*]," [Source Mr. Moussaoui] told his court-appointed lawyer Gerald T. Zerkin.

...
But, [Source Mr. Zerkin] pressed [Target Mr. Moussaoui], was it [– *not true*] that he told his lawyers earlier not to involve any Muslims in the defense, not to present any evidence that might persuade the jurors to spare his life?

...
[Source Zerkin] seemed to be trying to show the jurors that while [Target the defendant] is generally [+ *an honest individual*], his conduct shows [Target he] is [– *not stable mentally*], and thus [– *undeserving*] of [Target the ultimate punishment].

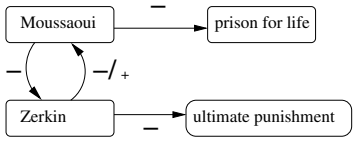

Figure 1: Example text containing opinions (above) and a summary of the opinions (below). Sources and targets of opinions are bracketed; opinion expressions are shown in italics and bracketed with associated polarity, either positive (+) or negative (-). The underlined phrase will be explained later in the paper.

2 Related Work

Work relevant to our problem can be split into three main areas – sentiment analysis, traditional noun phrase coreference resolution, and supervised and weakly supervised clustering. Related work in the former two areas is summarized briefly below. Supervised and weakly supervised clustering approaches are discussed in Section 4.

Sentiment analysis. Much of the relevant research in sentiment analysis addresses sentiment classification, a text categorization task of extracting opinion at the coarse-grained document level. The goal in sentiment classification is to assign to a document either positive ("thumbs up") or negative ("thumbs down") polarity (e.g. Das and Chen (2001), Pang et al. (2002), Turney (2002), Dave et al. (2003)). Other research has concentrated on analyzing fine-grained opinions at, or below, the sentence level. Recent work, for example, indicates that systems can be trained to recognize opinions and their polarity, strength, and sources to a reasonable degree of accuracy (e.g. Dave et al. (2003), Riloff and Wiebe (2003), Bethard et al. (2004), Wilson et al. (2004), Yu and Hatzivassiloglou (2003), Choi et al. (2005), Kim and Hovy (2005), Wiebe and Riloff (2005)). Our work extends research on fine-grained opinion extraction by augmenting the opinions with additional information that allows the creation of concise opinion summaries. In contrast to the opinion extracts produced by Pang and Lee (2004), our summaries are not text extracts, but rather explicitly identify and

characterize the relations between opinions and their sources.

Coreference resolution. Coreference resolution is a relatively well studied NLP problem (e.g. Morton (2000), Ng and Cardie (2002), Iida et al. (2003), McCallum and Wellner (2003)). Coreference resolution is defined as the problem of deciding which noun phrases in the text (*mentions*) refer to the same real world entities (*are coreferent*). Generally, successful approaches to coreference resolution have relied on supervised classification followed by clustering. For supervised classification these approaches learn a pairwise function to predict whether a pair of noun phrases is coreferent. Subsequently, when making coreference resolution decisions on unseen documents, the learnt pairwise NP coreference classifier is run, followed by a clustering step to produce the final clusters (coreference chains) of coreferent NPs. For both training and testing, coreference resolution algorithms rely on feature vectors for pairs of noun phrases that encode linguistic information about the NPs and their local context. Our general approach to source coreference resolution is inspired by the state-of-the-art performance of one such approach to coreference resolution, which relies on a rule learner and single-link clustering as described in Ng and Cardie (2002).

3 Source Coreference Resolution

In this section we introduce the problem of source coreference resolution in the context of opinion summarization and argue for the need for novel methods for the task.

The task of *source coreference resolution* is to decide which mentions of opinion sources refer to the same entity. Much like traditional coreference resolution, we employ a learning approach; however, our approach differs from traditional coreference resolution in its definition of the learning task. Motivated by the desire to utilize unlabeled examples (discussed later), we define training as an integrated task in which pairwise NP coreference decisions are learned together with the clustering function as opposed to treating each NP pair as a training example. Thus, our training phase takes as input a set of documents with manually annotated opinion sources together with coreference annotations for the sources; it outputs a classifier that can produce source coreference chains for previously unseen documents containing marked (manually or automatically) opinion sources. More specifically, the source coreference resolution training phase proceeds through the following steps:

1. **Source-to-NP mapping:** We preprocess each document by running a tokenizer, sentence splitter, POS tagger, parser, and an NP finder. Subsequently, we augment the set of NPs found by the NP finder with the help of a system for named entity detection. We then map the sources to the NPs. Since there is no one-to-one correspondence, we use a set of heuristics to create the mapping. More details about why heuristics are needed and the process used to map sources to NPs can be found in Stoyanov and Cardie (2006).

2. **Feature vector creation:** We extract a feature vector for every pair of NPs from the preprocessed corpus. We use the features introduced by Ng and Cardie (2002) for the task of coreference resolution.

3. **Classifier construction:** Using the feature vectors from step 2, we construct a training set containing one training example per document. Each training example consists of the feature vectors for all pairs of NPs in the document, including those that do not map to sources, together with the available coreference information for the *source noun phrases* (i.e. the noun phrases to which sources are mapped). The training instances are provided as input to a learning algorithm (see Section 5), which constructs a classifier that can take the instances associated with a new (previously unseen) document and produce a clustering over all NPs in the document.

The testing phase employs steps 1 and 2 as described above, but replaces step 3 by a straightforward application of the learnt classifier. Since we are interested in coreference information only for the source NPs, we simply discard the non-source NPs from the resulting clustering.

The approach to source coreference resolution described here would be identical to traditional coreference resolution when provided with training examples containing coreference information for all NPs. However, opinion corpora in general, and our corpus in particular, contain no coreference information about general NPs. Nevertheless, after manual sources are mapped to NPs in

step 1 above, our approach can rely on the available coreference information for the source NPs. Due to the high cost of coreference annotation, we desire methods that can work in the presence of only this limited amount of coreference information.

A possible workaround the absence of full NP coreference information is to train a traditional coreference system only on the labeled part of the data (indeed that is one of the baselines against which we compare). However, we believe that an effective approach to source coreference resolution has to utilize the unlabeled noun phrases because links between sources might be realized through non-source mentions. This problem is illustrated in figure 1. The underlined *Moussaoui* is coreferent with all of the Moussaoui references marked as sources, but, because it is used in an objective sentence rather than as the source of an opinion, the reference would be omitted from the *Moussaoui* source chain. Unfortunately, this proper noun phrase might be critical in establishing the coreference of the final source reference *he* with the other mentions of the source *Moussaoui*.

As mentioned previously, in order to utilize the unlabeled data, our approach differs from traditional coreference resolution, which uses NP pairs as training instances. We instead follow the framework of supervised clustering (Finley and Joachims, 2005; Li and Roth, 2005) and consider each document as a training example. As in supervised clustering, this framework has the additional advantage that the learning algorithm can consider the clustering algorithm when making decisions about pairwise classification, which could lead to improvements in the classifier. In the next section we describe our approach to classifier construction for step 3 and compare our problem to traditional weakly supervised clustering, characterizing it as an instance of the novel problem of partially supervised clustering.

4 Partially Supervised Clustering

In our desire to perform effective source coreference resolution we arrive at the following learning problem – the learning algorithm is presented with a set of partially specified examples of clusterings and acquires a function that can cluster accurately an unseen set of items, while taking advantage of the unlabeled information in the examples.

This setting is to be contrasted with semi-supervised clustering (or clustering with constraints), which has received much research attention (e.g. Demiriz et al. (1999), Wagstaff and Cardie (2000), Basu (2005), Davidson and Ravi (2005)). Semi-supervised clustering can be defined as the problem of clustering a set of items in the presence of limited supervisory information such as pairwise constraints (e.g. two items must/cannot be in the same cluster) or labeled points. In contrast to our setting, in the semi-supervised case there is no training phase – the algorithm receives all examples (labeled and unlabeled) at the same time together with some distance or cost function and attempts to find a clustering that optimizes a given measure (usually based on the distance or cost function).

Source coreference resolution might alternatively be approached as a supervised clustering problem. Traditionally, approaches to supervised clustering have treated the pairwise link decisions as a classification problem. These approaches first learn a distance metric that optimizes the pairwise decisions; and then follow the pairwise classification with a clustering step. However, these traditional approaches have no obvious way of utilizing the available unlabeled information.

In contrast, we follow recent approaches to supervised clustering that propose ways to learn the distance measure in the context of the clustering decisions (Li and Roth, 2005; Finley and Joachims, 2005; McCallum and Wellner, 2003). This provides two advantages for the problem of source coreference resolution. First, it allows the algorithm to take advantage of the complexity of the rich structural dependencies introduced by the clustering problem. Viewed traditionally as a hurdle, the structural complexity of clustering may be beneficial in the partially supervised case. We believe that provided with a few partially specified clustering examples, an algorithm might be able to generalize from the structural dependencies to infer correctly the whole clustering of the items. In addition, considering pairwise decisions in the context of the clustering can arguably lead to more accurate classifiers.

Unfortunately, none of the supervised clustering approaches is readily applicable to the partially supervised case. However, by adapting the formal supervised clustering definition, which we do next, we can develop approaches to partially supervised clustering that take advantage of the un-

labeled portions of the data.

Formal definition. For partially supervised clustering we extend the formal definition of supervised clustering given by Finley and Joachims (2005). In the fully supervised setting, an algorithm is given a set S of n training examples $(x_1, y_1), ..., (x_n, y_n) \in X \times Y$, where X is the set of all possible sets of items and Y is the set of all possible clusterings of these sets. For a training example (x, y), $x = \{x_1, x_2, ..., x_k\}$ is a set of k items and $y = \{y_1, y_2, ..., y_r\}$ is a clustering of the items in x with each $y_i \subseteq x$. Additionally, each item can be in no more than one cluster ($\forall i, j . y_i \cap y_j = \emptyset$) and in the fully supervised case each item is in at least one cluster ($x = \bigcup y_i$). The goal of the learning algorithm is to acquire a function $h : X \rightarrow Y$ that can accurately cluster a (previously unseen) set of items.

In the context of source coreference resolution the training set contains one example for each document. The items in each training example are the NPs and the clustering over the items is the equivalence relation defined by the coreference information. For source coreference resolution, however, clustering information is unavailable for the non-source NPs. Thus, to be able to deal with this unlabeled component of the data we arrive to the setting of partially supervised clustering, in which we relax the condition that each item is in at least one cluster ($x = \bigcup y_i$) and replace it with the condition $x \supseteq \bigcup y_i$. The items with no linking information (items in $x \setminus \bigcup y_i$) constitute the unlabeled (unsupervised) component of the partially supervised clustering.

5 Structured Rule Learner

We develop a novel method for partially supervised clustering, which is motivated by the success of a rule learner (RIPPER) for coreference resolution (Ng and Cardie, 2002). We extend RIPPER so that it can learn rules in the context of single-link clustering, which both suits our task (i.e. pronouns link to their single antecedent) and has exhibited good performance for coreference resolution (Ng and Cardie, 2002). We begin with a brief overview of RIPPER followed by a description of the modifications that we implemented. For ease of presentation, we assume that we are in the fully supervised case. We end this section by describing the changes for the partially supervised case.

```
procedure StRip(TrainData){
  GrowData, PruneData = Split(TrainData);
  //Keep instances from the same document together
  while(there are positive uncovered instances) {
    r = growRule(GrowData);
    r = pruneRule(r, PruneData);
    DL = relativeDL(Ruleset);
    if(DL ≤ minDL + d bits)
      Ruleset.add(r);
      Mark examples covered by r as +;
    else
      exit loop with Ruleset
  }
}
procedure growRule(growData){
  r = empty rule;
  for(every unused feature f){
    if (f is nominal feature) {
      for(every possible value v of f) {
        mark all instances that have values of v for f with +;
        compute the transitive closure of the positive instances
        //(including instances marked + from previous rules);
        compute the infoGain for the future/value combination;
      }
    } else{ //Numeric feature
      create one bag for each feature value and split the instances into bags;
      do a forward and a backward pass over the bags keeping a running
      clustering and compute the information gain for each value;
    }
  }
  add the future/value pair with the best infoGain to r;
  growData = growData - all negative instances;
  return r;
}
procedure pruneRule(r, pruneData){
  for(all antecedents a in the rule){
    apply all antecedents in r up to a to pruneData;
    compute the transitive closure of the positive instances;
    compute A(a) – the accuracy of the rule up to antecedent a;
  }
  Remove all antecedents after the antecedent for which A(a) is maximum.
}
```

Figure 2: The StRip algorithm. Additions to RIPPER are shown in bold.

5.1 The RIPPER Algorithm

RIPPER (for Repeated Incremental Pruning to Produce Error Reduction) was introduced by Cohen (1995) as an extension of an existing rule induction algorithm. Cohen (1995) showed that RIPPER produces error rates competitive with C4.5, while exhibiting better running times. RIPPER consists of two phases – a ruleset is grown and then optimized.

The ruleset creation phase begins by randomly splitting the training data into a rule-growing set (2/3 of the training data) and a pruning set (the remaining 1/3). A rule is then grown on the former set by repeatedly adding the *antecedent* (the feature value test) with the largest information gain until the accuracy of the rule becomes 1.0 or there are no remaining potential antecedents. Next the rule is applied to the pruning data and any rule-final sequence that reduces the accuracy of the rule is removed.

The optimization phase uses the full training

set to first grow a replacement rule and a revised rule for each rule in the ruleset. For each rule, the algorithm then considers the original rule, the replacement rule, and the revised rule, and keeps the rule with the smallest description length in the context of the ruleset. After all rules are considered, RIPPER attempts to grow residual rules that cover data not already covered by the ruleset. Finally, RIPPER deletes any rules from the ruleset that reduce the overall minimum description length of the data plus the ruleset. RIPPER performs two rounds of this optimization phase.

5.2 The StRip Algorithm

The property of partially supervised clustering that we want to explore is the structured nature of the decisions. That is, each decision of whether two items (say a and b) belong to the same cluster has an implication for all items a' that belong to a's cluster and all items b' that belong to b's cluster.

We target modifications to RIPPER that will allow StRip (for Structured RIPPER) to learn rules that produce good clusterings in the context of single-link clustering. We extend RIPPER so that every time it makes a decision about a rule, it considers the effect of the rule on the overall clustering of items (as opposed to considering the instances that the rule classifies as positive/negative in isolation). More precisely, we precede every computation of rule performance (e.g. information gain or description length) by a transitive closure (i.e. single link clustering) of the data w.r.t. to the pairwise classifications. Following the transitive closure, all pairs of items that are in the same cluster are considered covered by the rule for performance computation.

The StRip algorithm is given in figure 2, with modifications to the original RIPPER algorithm shown in bold. Due to space limitations the optimization stage of the algorithm is omitted. Our modifications to the optimization stage of RIPPER are in the spirit of the rest of the StRip algorithm.

Partially supervised case. So far we described StRip only for the fully supervised case. We use a very simple modification to handle the partially supervised setting: we exclude the unlabeled pairs when computing the performance of the rules. Thus, the unlabeled items do not count as correct or incorrect classifications when acquiring or pruning a rule, although they do participate in the transitive closure. Links in the unlabeled data are inferred entirely through the indirect links between items in the labeled component that they introduce. In the example of figure 1, the two problematic unlabeled links are the link between the source mention "he" and the underlined non-source NP "Mr. Moussaoui" and the link between the underlined "Mr. Moussaoui" to any source mention of *Moussaoui*. While StRip will not reward any rule (or rule set) that covers these two links directly, such rules will be rewarded indirectly since they put the source *he* in the chain for the source *Moussaoui*.

StRip running time. StRip's running time is generally comparable to that of RIPPER. We compute transitive closure by using a Union-Find structure, which runs in time $O(log^* n)$, which for practical purposes can be considered linear ($O(n)$) [3]. However, when computing the best information gain for a nominal feature, StRip has to make a pass over the data for each value that the feature takes, while RIPPER can split the data into bags and perform the computation in one pass.

6 Evaluation and Results

This section describes the source coreference data set, the baselines, our implementation of StRip, and the results of our experiments.

6.1 Data set

For evaluation we use the MPQA corpus (Wiebe et al., 2005).[4] The corpus consists of 535 documents from the world press. All documents in the collection are manually annotated with phrase-level opinion information following the annotation scheme of Wiebe et al. (2005). Discussion of the annotation scheme is beyond the scope of this paper; for our purposes it suffices to say that the annotations include the source of each opinion and coreference information for the sources (e.g. source coreference chains). The corpus contains no additional noun phrase coreference information.

For our experiments, we randomly split the data set into a training set consisting of 400 documents and a test set consisting of the remaining 135 documents. We use the same test set for all experi-

[3] For the transitive closure, n is the number of items in a document, which is $O(\sqrt{k})$, where k is the number of NP pairs. Thus, transitive closure is sublinear in the number of training instances.

[4] The MPQA corpus is available at `http://nrrc.mitre.org/NRRC/publications.htm`.

ments, although some learning runs were trained on 200 training documents (see next Subsection). The test set contains a total of 4736 source NPs (average of 35.34 source NPs per document) split into 1710 total source NP chains (average of 12.76 chains per document) for an average of 2.77 source NPs per chain.

6.2 Implementation

We implemented the StRip algorithm by modifying JRip – the java implementation of RIPPER included in the WEKA toolkit (Witten and Frank, 2000). The WEKA implementation follows the original RIPPER specification. We changed the implementation to incorporate the modifications suggested by the StRip algorithm; we also modified the underlying data representations and data handling techniques for efficiency. Also due to efficiency considerations, we train StRip only on the 200-document training set.

6.3 Competitive baselines

We compare the results of the new method to three fully supervised baseline systems, each of which employs the same traditional coreference resolution approach. In particular, we use the aforementioned algorithm proposed by Ng and Cardie (2002), which combines a pairwise NP coreference classifier with single-link clustering.

For one baseline, we train the coreference resolution algorithm on the *MPQA src* corpus — the labeled portion of the MPQA corpus (i.e. NPs from the source coreference chains) with unlabeled instances removed.

The second and third baselines investigate whether the source coreference resolution task can benefit from NP coreference resolution training data *from a different domain*. Thus, we train the traditional coreference resolution algorithm on the *MUC6* and *MUC7* coreference-annotated corpora[5] that contain documents similar in style to those in the MPQA corpus (e.g. newspaper articles), but emanate from different domains.

For all baselines we targeted the best possible systems by trying two pairwise NP classifiers (RIPPER and an SVM in the SVM^{light} implementation (Joachims, 1998)), many different parameter settings for the classifiers, two different feature sets, two different training set sizes (the full training set and a smaller training set consisting of half of the documents selected at random), and three different instance selection algorithms[6]. This variety of classifier and training data settings was motivated by reported differences in performance of coreference resolution approaches w.r.t. these variations (Ng and Cardie, 2002). More details on the different parameter settings and instance selection algorithms as well as trends in the performance of different settings can be found in Stoyanov and Cardie (2006). In the experiments below we report the best performance of each of the two learning algorithms on the MPQA test data.

6.4 Evaluation

In addition to the baselines described above, we evaluate StRip both with and without unlabeled data. That is, we train on the MPQA corpus StRip using either all NPs or just opinion source NPs.

We use the B^3 (Bagga and Baldwin, 1998) evaluation measure as well as precision, recall, and F1 measured on the (positive) pairwise decisions. B^3 is a measure widely used for evaluating coreference resolution algorithms. The measure computes the precision and recall for each NP mention in a document, and then averages them to produce combined results for the entire output. More precisely, given a mention i that has been assigned to chain c_i, the precision for mention i is defined as the number of correctly identified mentions in c_i divided by the total number of mentions in c_i. Recall for i is defined as the number of correctly identified mentions in c_i divided by the number of mentions in the gold standard chain for i.

Results are shown in Table 1. The first six rows of results correspond to the fully supervised baseline systems trained on different corpora — *MUC6*, *MUC7*, and *MPQA src*. The seventh row of results shows the performance of StRip using only labeled data. The final row of the table shows the results for partially supervised learning **with** unlabeled data. The table lists results from the best performing run for each algorithm.

Performance among the baselines trained on the *MUC* data is comparable. However, the two baseline runs trained on the *MPQA src* corpus (i.e. results rows five and six) show slightly better performance on the B^3 metric than the baselines trained

[5]We train each baseline using both the development set and the test set from the corresponding MUC corpus.

[6]The goal of the instance selection algorithms is to balance the data, which contains many more negative than positive instances

ML Framework	Training set	Classifier	B^3	precision	recall	F1
Fully supervised	MUC6	SVM	81.2	*72.6*	*52.5*	60.9
		RIPPER	80.7	*57.4*	*63.5*	60.3
	MUC7	SVM	81.7	*65.6*	*55.9*	60.4
		RIPPER	79.7	*71.6*	*48.5*	57.9
	MPQA src	SVM	81.8	*57.5*	*62.9*	60.2
		RIPPER	81.8	*72.0*	*52.5*	60.6
		StRip	82.3	*76.5*	*56.1*	64.6
Partially supervised	MPQA all	StRip	**83.2**	*77.1*	*59.4*	**67.1**

Table 1: Results for Source Coreference. *MPQA src* stands for the MPQA corpus limited to only source NPs, while *MPQA full* contains the unlabeled NPs.

on the *MUC* data, which indicates that for our task the similarity of the documents in the training and test sets appears to be more important than the presence of complete supervisory information. (Improvements over the RIPPER runs trained on the MUC corpora are statistically significant[7], while improvements over the SVM runs are not.)

Table 1 also shows that StRip outperforms the baselines on both performance metrics. StRip's performance is better than the baselines when trained on *MPQA src* (improvement not statistically significant, $p > 0.20$) and even better when trained on the full MPQA corpus, which includes the unlabeled NPs (improvement over the baselines and the former StRip run statistically significant). These results confirm our hypothesis that StRip improves due to two factors: first, considering pairwise decisions in the context of the clustering function leads to improvements in the classifier; and, second, StRip can take advantage of the unlabeled portion of the data.

StRip's performance is all the more impressive considering the strength of the SVM and RIPPER baselines, which which represent the best runs across the 336 different parameter settings tested for SVM^{light} and 144 different settings tested for RIPPER. In contrast, all four of the StRip runs using the full MPQA corpus (we vary the loss ratio for false positive/false negative cost) outperform those baselines.

7 Future Work

Source coreference resolution is only one aspect of opinion summarization. Additionally, an opinion summarization system will need to handle the closely related task of target coreference resolution in order to cluster targets of opinions[8] and combine multiple conflicting opinions from a source to the same targets. Furthermore, a fully automatic opinion summarizer requires automatic source and opinion extractors. While we anticipate that target coreference resolution will be subject to error rates similar to those of source coreference resolution, incorporating these imperfect opinions and sources will further impair the performance of the opinion summarizer. We are not aware of any measure that can be directly used to assess the goodness of opinion summaries, but plan to develop such in future work in conjunction with the development of methods for creating opinion summaries completely automatically. The evaluation metrics will likely have to depend on the task for which the summaries are used.

A limitation of our approach to partially supervised clustering is that we do not directly optimize for the performance measure (e.g. B^3). Other efforts in the area of supervised clustering (Finley and Joachims, 2005; Li and Roth, 2005) have suggested ways to learn distance measures that can optimize directly for a desired performance measure. We plan to investigate algorithms that can directly optimize for complex measures (such as B^3) for the problem of partially supervised clustering. Unfortunately, a measure as complex as B^3 makes extending existing approaches far from trivial due to the difficulty of establishing the connection between individual pairwise decisions (the distance metric) and the score of the clustering algorithm.

Acknowledgements

The authors would like to thank Vincent Ng and Art Munson for providing coreference resolution

[7] Statistical significance is measured using both a 2-tailed paired t-test and the Wilcoxon matched-pairs signed-ranks test ($p < 0.05$). The two tests agreed on all significance judgements, so we will not report them separately.

[8] We did not tackle the task of target coreference resolution in this paper because the MPQA corpus did not contain target annotations at the time of publication.

code, members of the Cornell NLP group (especially Yejin Choi and Art Munson) for many helpful discussions, and the anonymous reviewers for their insightful comments. This work was supported by the Advanced Research and Development Activity (ARDA), by NSF Grants IIS-0535099 and IIS-0208028, by gifts from Google and the Xerox Foundation, and by an NSF Graduate Research Fellowship to the first author.

References

A. Bagga and B. Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *In Proceedings of COLING/ACL*.

S. Basu. 2005. *Semi-supervised Clustering: Probabilistic Models, Algorithms and Experiments*. Ph.D. thesis, Department of Computer Sciences, UT at Austin.

S. Bethard, H. Yu, A. Thornton, V. Hativassiloglou, and D. Jurafsky. 2004. Automatic extraction of opinion propositions and their holders. In *2004 AAAI Spring Symposium on Exploring Attitude and Affect in Text*.

Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. 2005. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of EMNLP*.

C. Coglianese. 2004. E-rulemaking: Information technology and regulatory policy: New directions in digital government research. Technical report, Harvard University, J. F. Kennedy School of Government.

W. Cohen. 1995. Fast effective rule induction. In *Proceedings of ICML*.

S. Das and M. Chen. 2001. Yahoo for amazon: Extracting market sentiment from stock message boards. In *Proceedings of APFAAC*.

K. Dave, S. Lawrence, and D. Pennock. 2003. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of IWWWC*.

I. Davidson and S. Ravi. 2005. Clustering with constraints: Feasibility issues and the k-means algorithm. In *Proceedings of SDM*.

A. Demiriz, K. P. Bennett, and M. J. Embrechts. 1999. Semi-supervised clustering using genetic algorithms. In *Proceeding of ANNIE*.

T. Finley and T. Joachims. 2005. Supervised clustering with support vector machines. In *Proceedings of ICML*.

R. Iida, K. Inui, H. Takamura, and Y. Matsumoto. 2003. Incorporating contextual cues in trainable models for coreference resolution. In *Proceedings of the EACL Workshop on The Computational Treatment of Anaphora*.

T. Joachims. 1998. Making large-scale support vector machine learning practical. In A. Smola B. Schölkopf, C. Burges, editor, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA.

S. Kim and E. Hovy. 2005. Identifying opinion holders for question answering in opinion texts. In *Proceedings of AAAI Workshop on Question Answering in Restricted Domains*.

X. Li and D. Roth. 2005. Discriminative training of clustering functions: Theory and experiments with entity identification. In *Proceedings of CoNLL*.

A. McCallum and B. Wellner. 2003. Toward conditional models of identity uncertainty with application to proper noun coreference. In *Proceedings of the IJCAI Workshop on Information Integration on the Web*.

T. Morton. 2000. Coreference for NLP applications. In *Proceedings of ACL*.

V. Ng and C. Cardie. 2002. Improving machine learning approaches to coreference resolution. In *In Proceedings of ACL*.

B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of ACL*.

B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of EMNLP*.

E. Riloff and J. Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceesings of EMNLP*.

E. Riloff, J. Wiebe, and W. Phillips. 2005. Exploiting subjectivity classification to improve information extraction. In *Proceedings of AAAI*.

V. Stoyanov and C. Cardie. 2006. Toward opinion summarization: Linking the sources. In *Proceedings of the ACL Workshop on Sentiment and Subjectivity in Text*.

V. Stoyanov, C. Cardie, and J. Wiebe. 2005. Multi-Perspective question answering using the OpQA corpus. In *Proceedings of EMNLP*.

P. Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL*.

K. Wagstaff and C. Cardie. 2000. Clustering with instance-level constraints. In *Proceedings of the 17-th National Conference on Artificial Intelligence and 12-th Conference on Innovative Applications of Artificial Intelligence*.

J. Wiebe and E. Riloff. 2005. Creating subjective and objective sentence classifiers from unannotated texts. In *Proceedings of CICLing*.

J. Wiebe, T. Wilson, and C. Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 1(2).

T. Wilson, J. Wiebe, and R. Hwa. 2004. Just how mad are you? Finding strong and weak opinion clauses. In *Proceedings of AAAI*.

I.H. Witten and E. Frank. 2000. *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco.

H. Yu and V. Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of EMNLP*.