Iterative Constrained Clustering for Subjectivity Word Sense Disambiguation

Cem Akkaya, Janyce Wiebe
University of Pittsburgh
Pittsburgh PA, 15260, USA
{cem,wiebe}@cs.pitt.edu

Rada Mihalcea
University of North Texas
Denton TX, 76207, USA
rada@cs.unt.edu

Abstract

Subjectivity word sense disambiguation (SWSD) is a supervised and application-specific word sense disambiguation task disambiguating between subjective and objective senses of a word. Not surprisingly, SWSD suffers from the knowledge acquisition bottleneck. In this work, we use a "cluster and label" strategy to generate labeled data for SWSD semi-automatically. We define a new algorithm called *Iterative Constrained Clustering (ICC)* to improve the clustering purity and, as a result, the quality of the generated data. Our experiments show that the SWSD classifiers trained on the ICC generated data by requiring only 59% of the labels can achieve the same performance as the classifiers trained on the full dataset.

1 Introduction

Subjectivity lexicons (e.g., (Turney, 2002; Whitelaw et al., 2005; Riloff and Wiebe, 2003; Yu and Hatzivassiloglou, 2003; Kim and Hovy, 2004; Bloom et al., 2007; Andreevskaia and Bergler, 2008; Agarwal et al., 2009)) play an important role in opinion, sentiment, and subjectivity analysis. These systems typically look for the presence of clues in text. Recently, in (Akkaya et al., 2009), we showed that subjectivity clues are fairly ambiguous as to whether they express subjectivity or not – words in such lexicons may have both subjective and objective usages. We call this problem subjectivity sense ambiguity. Consider the following sentence containing the clue "attack":

(1) He was **attacked** by Milosevic for attempting to carve out a new party from the Socialists.

Knowing that "attack" is a subjectivity clue with negative polarity will help a system recognize the negative sentiment in the sentence. But for (2), the same information is simply misleading, because the clue is used with an objective meaning.

(2) A new treatment based on training T-cells to **attack** cancerous cells ...

Any opinion analysis system which relies on a subjectivity lexicon will be misled by subjectivity clues used with objective senses (false hits). In (Akkaya et al., 2009), we introduced the task, *Subjectivity Word Sense Disambiguation*, which is to automatically determine which word instances in a corpus are being used with subjective senses, and which are being used with objective senses. SWSD can be considered as a coarse-grained and application-specific word sense disambiguation task. We showed that sense subjectivity information about clues can be fed to subjectivity and sentiment analysis resulting in substantial improvement for both subjectivity and sentiment analysis by avoiding false hits.

Although SWSD is a promising tool, it suffers from the knowledge acquisition bottleneck. SWSD is defined as a supervised task, and follows a targeted approach common in the WSD literature for performance reasons. This means, for each target clue, a different classifier is trained requiring separate training data for each target clue. It is expensive and time-consuming to obtain annotated datasets to train SWSD classifiers limiting scalability. As a countermeasure, in (Akkaya et al., 2011), we showed that non-expert annotations collected through Amazon Mechanical Turk (MTurk) can replace expert annotations successfully and might be used to apply SWSD on a large scale.

Although non-expert annotations are cheap and fast, they still incur some cost. In this work, we aim to reduce the human annotation effort needed to generate the same amount of subjectivity sense tagged data by using a "cluster and label" strategy. We hypothesize that we can obtain large sets of labeled data by labelling clusters of instances of a target word instead of single instances.

The main contribution of this work is a novel constrained clustering algorithm called *Iterative Constrained Clustering (ICC)* utilizing an active constraint selection strategy. A secondary contribution is a mixed word representation that is a combination of previously proposed context representations. We show that a "cluster and label" strategy relying on these two proposed components generates training data of good purity. The resulting data has sufficient purity to train reliable SWSD classifiers. SWSD classifiers trained on only 59% of the data achieve the same performance as classifiers trained on 100% of the data, resulting in a significant reduction in the annotation effort. Our results take SWSD another step closer to large scale application.

2 Cluster and Label

Our approach is inspired by a method lexicographers commonly employ to create sense inventories, where they create inventories based on evidence found in corpora. They use concordance information to mine frequent usage patterns. (Kilgarriff, 1997) describes this process in detail. A lexicographer collects usages of a word in corpora and groups them into coherent sets. The instances in a set should have more in common with each other than with the instances in other sets, according to the criteria the lexicographer considers. After generating the sets, the lexicographer codes each set as a dictionary definition based on the common attributes of the instances. Our goal is similar. Instead of generating dictionary definitions, we are only interested in generating coherent sets of usages of a word, so that we can label each induced set – with its instances – to obtain labeled data for SWSD. Our high-level grouping criterion is that the instances in a cluster should be similar subjective (objective) usages of the word.

Training data for an SWSD classifier consists of instances of the target word tagged as having a subjective sense (*S*) or an objective sense (*O*) (*subjectivity sense tagged data*). We train a different SWSD classifier for each target word as in (Akkaya et al., 2009). Thus, we need a different training dataset for each target word. Our ultimate goal is to reduce the human annotation effort required to create training data for SWSD classifiers. For this purpose, we utilize a *"cluster and label"* strategy relying on context clustering. Each instance of a word is represented as a feature vector (i.e., a context vector). The annotation process has the following steps: (1) cluster the context vectors of word instances, (2) label the induced clusters as *S* or *O*, (3) propagate the given label to all instances in a cluster.

The induced clusters represent different usage patterns of a word. Thus, we build more than two clusters, even though SWSD is a binary task. This implies that two different instances of a word can both be subjective, but end up in different clusters, if they are different usages of the word.

Since we are labelling clusters as a whole, we will introduce noise in the labeled data. Thus, in developing the clustering process, we need to minimize that noise and find as pure clusters as possible.

The first step is to define the context representation of the instances. This is addressed in Section 3. Then, we turn in Section 4.2 to the clustering process itself.

To evaluate our "cluster and label" strategy, we use two gold standard subjectivity sense tagged datasets. [1]. The first one is called senSWSD generated in (Akkaya et al., 2009) and the second one is called mturkSWSD generated in (Akkaya et al., 2011). They consist of subjectivity sense tagged data for disjoint sets of 39 and 90 words, respectively. In this paper, we opt to use the smaller dataset senSWSD as our development set, on which we evaluate various context representations (in Section 3) and our proposed constrained clustering algorithm (in Section 4.2). Then, on mturkSWSD, we evaluate the quality of semi-automatically generated data for SWSD classification (in Section 4.3.2).

3 Context Representations

There has been much work on context representations of words for various NLP tasks. Clustering word instances in order to discriminate senses of a word is called *Word Sense Discrimination*. Context representations for this task rely on two main types of models: distributional semantic models (DSM) and feature-based models.

[1]Available at `http://mpqa.cs.pitt.edu/corpora`

(Schutze, 1998), which is still a competitive model for word-sense discrimination by context clustering, relies on a distributional semantic model (DSM) (Turney and Pantel, 2010; Sahlgren, 2006; Bullinaria and Levy, 2007). A DSM is usually a word-to-word co-occurrence matrix – also called *semantic space* – such that each row represents the distribution of a target word in a large text corpus. Each row gives the semantic signature of a word, which is basically a high dimensional numeric vector. Note that this high dimensional vector represents word types, not word tokens. Thus, it cannot model a word instance in context. For token-based treatment, (Schutze, 1998) utilizes a second-order representation by averaging co-occurrence vectors of the words (corresponding to rows of the co-occurrence matrix) that occur in that particular context. It is important to note that (Schutze, 1998) uses an additive model for compositional representation. Recently, in (Akkaya et al., 2012), we found that a DSM built using multiplicative composition – proposed by (Mitchell and Lapata, 2010) for a different task – gives better performance than the model described by (Schutze, 1998).

We test both methods in this paper, using the same semantic space. The space is built from a corpus consisting of 120 million tokens. The rows of the space correspond to word forms and the columns correspond to word lemmas present in the corpus. We adopt the parameters for our semantic space from (Mitchell and Lapata, 2010): window size of 10 and dimension size of 2000 (i.e., the 2000 most frequent lemmas). We do not filter out stop words, since they have been shown to be useful for various semantic similarity tasks in (Bullinaria and Levy, 2007). We use positive point-wise mutual information to compute values of the vector components, which has also been shown to be favourable in (Bullinaria and Levy, 2007).

Purandere and Pedersen is the prominent representative of feature-based models. (Purandare and Pedersen, 2004) creates context vectors from local feature representations similar to the feature vectors found in supervised WSD. In this work, we use the following features from (Mihalcea, 2002) to build the local feature representation: (1) the target word itself and its part of speech, (2) surrounding context of 3 words and their part of speech, (3) the head of the noun phrase, (4) the first noun and verb before the target word, (5) the first noun and verb after the target word.

	skew	local	dsm_add	dsm_mul	mix_rep
average	79.90	80.50	80.50	83.53	**85.23**
appear-v	53.83	54.85	54.85	57.40	69.39
fine-a	70.07	72.26	70.07	74.45	75.18
interest-n	54.41	54.78	55.88	81.62	81.62
restraint-n	70.45	71.97	75.00	71.21	81.82

Table 1: Evaluation of Various Context Representations

3.1 Evaluation of Context Representations

In this section, we evaluate context representations for the context clustering task on the subjectivity sense tagged data, senSWSD. The evaluation is done separately for each word.

We use the same clustering algorithm for all context representations: agglomerative hierarchical clustering with average linkage criteria. In all our experiments throughout the paper, we fix the cluster size to 7 as it is done in (Purandare and Pedersen, 2004). We think that is reasonable number since SENSEVAL III reports that the average number of senses per word is 6.47. We choose *cluster purity* as our evaluation metric. To compute cluster purity, we assign each cluster to a sense label, which is the most frequent one in the cluster. The number of the correctly assigned instances divided by the number of all the clustered instances gives us cluster purity.

Row 1 of Table 1 holds the cumulative results over all the words in senSWSD (micro averages). The table also reports detailed results for 4 sample selected words from senSWSD. *skew* stands for the percentage of the most frequent label. *dsm_add* is the representation based on (Schutze, 1998), *dsm_mul* stands for the representation as described in (Akkaya et al., 2012) and *local_features* is the local feature representation based on (Purandare and Pedersen, 2004). The results show that among *dsm_mul*, *dsm_add*, and *local_features*; *dsm_mul* performs the best.

When we look at the context clustering results for single words separately, we observe that the performance of different representations vary. There is not a single winner among all words. Thus, perhaps choosing one single representation for all the words is not optimal. Having that in mind, we try merging the *dsm_mul* and *local_features* representations. We leave out *dsm_add* representation, since both *dsm_mul* and *dsm_add* rely on the same type of semantic information (i.e., a DSM). We hypothesize that the two representations, one relying on a semantic space and the other relying on local WSD features, may complement each other.

To merge the representations, we concatenate the two feature vectors into one. First, however, we normalize each vector to unit length, since the individual vectors have different scales and would have unequal contribution, otherwise. We call this mixed representation *mix_rep*.

In Table 1, we see that, overall, *mix_rep* performs better than all the other representations. The improvement is statistically significant at the $p < .05$ level on a paired t-test. We observe that, even when *mix_rep* does not perform the best, it is never bad. *mix_rep* is the winner or ties for the winner for 25 out of 39 words. This number is 13, 13, and 15 for *dsm_add*, *dsm_mul* and *local_features*, respectively. For the words for which *mix_rep* is not the winner, it is, on average, 1.47 points lower than the winner. This number is 4.22, 6.83, and 7.07 for the others. The results provide evidence that *mix_rep* is consistently good and reliable. Thus, in our experiments, *mix_rep* will be our choice as the context representation.

4 Clustering Process

We now turn to the clustering process. In a "cluster and label" strategy, in order to be able to label clusters, we need to annotate some of the instances in each cluster. Then, we can accept the majority label found in a cluster as its label. Thus, some manual labelling is required, preferably a small amount.

We propose to provide this small amount of annotated data prior to clustering, and then perform semi-supervised clustering. This way the provided labels will guide the clustering algorithm to generate the clusters that are more suitable for our end task, namely clusters where subjective and objective instances are grouped together.

4.1 Constrained Clustering

Constrained clustering (Grira et al., 2004) also known as *semi-supervised clustering* is a recent development in the clustering literature. In addition to the similarity information required by unsupervised clustering, constrained clustering requires pairwise constraints. There are two types of constraints: (1) must-link and (2) cannot-link constraints. A must-link constraint dictates that two instances should be in the same cluster and a cannot-link dictates that two instances should not be in the same cluster. In this work, we only consider cannot-links, because of the definition of our SWSD task. Two instances sharing the same label do not need to be in the same cluster, since the induced clusters represent different usage patterns of a word. For example, two instances labeled S need not be similar to each other. They can be different usages, both having a subjective meaning. On the other hand, if two instances are labeled having opposing labels, we do not want them to be in the same cluster. Thus, we utilize cannot-links but not must-links.

Constraints can be obtained from domain knowledge or from available instance labels. In our work, constraints are generated from instance labels. Each instance pair with opposing labels is considered to be cannot-linked.

There are two general strategies to incorporate constraints into clustering. The first is to adapt the similarity between instances (Xing et al., 2002; Klein et al., 2002) by adjusting the underlying distance metric. The main idea is to make the distance between must-linked instances – their neighbourhoods – smaller and the distance between cannot-linked instances – their neighbourhoods – larger. The second strategy is modifying the clustering algorithm itself so that search is biased towards a partitioning for which the constraints hold (Wagstaff and Cardie, 2000; Basu et al., 2002; Demiriz et al., 1999).

Our proposed constrained clustering method relies on some ideas from (Klein et al., 2002). Thus, we explain it in more detail. (Klein et al., 2002) utilizes agglomerative hierarchical clustering with complete-linkage. The algorithm imposes constraints by changing the distance matrix according to the given constraints. The distances between must-linked instances are set to 0. That is not enough by itself, since if two instances are must-linked, other instances close to them should also get closer to each other. This means there is a need to propagate the constraints. This is done by calculating the shortest path between all the instances and updating the distance matrix accordingly. To impose cannot-links, the distance between two cannot-linked instances is set to some large number. The complete-linkage property indirectly propagates the cannot-link constraints, since it will not allow two clusters to be merged if they contain instances that are cannot-linked.

Although previous work report on average substantial improvement in the clustering purity, (Davidson et al., 2006) shows that even if the constraints are generated from gold-standard data, some constraint sets can decrease clustering purity. The results vary significantly depending on the specific set of constraints used. To our knowledge, there have been two approaches for selecting informative constraint sets (Basu et al., 2004; Klein et al., 2002). The method described in (Basu et al., 2004) uses the farthest-first traversal scheme. That strategy is not suitable in our setting, since we have only two labels. After selecting just one instance from both labels, this method becomes the same as random selection. The strategy described in (Klein et al., 2002) is more general. At first, the hierarchical clustering algorithm follows in a unconstrained fashion until some moderate number of clusters are remaining. Then, the algorithm starts to request constraints between roots whenever two clusters are merged.

4.2 Iterative Constrained Clustering

Our proposed algorithm is closely related to (Klein et al., 2002). We share the same backbone: (1) the agglomerative hierarchical clustering with complete-linkage and (2) the mechanism to impose cannot-link constraints described in Section 4.1. For our algorithm, we implement a second mechanism for imposing constraints proposed by (Xing et al., 2002) (Section 4.2.1) and use both mechanisms in combination. We also propose a novel constraint selection method (Section 4.2.2).

4.2.1 Imposing Constraints

(Klein et al., 2002) imposes cannot-link constraints by adjusting the distance between cannot-linked pairs heuristically and by relying on complete linkage for propagation. Although this approach was shown to be effective, we believe it does not make full use of the provided constraints. We believe that learning a new distance metric will result in more reliable distance estimates between all instances. For this purpose, we learn a Mahalanobis distance function following the method described in (Davis et al., 2007). (Davis et al., 2007) formulate the problem of distance metric learning as minimizing the differential relative entropy between two multivariate Gaussians under constraints. Note that using distance metric learning for imposing constraints was previously proposed by (Xing et al., 2002). (Xing et al., 2002) pose metric learning as a convex optimization problem. The reason we choose the metric learning method (Davis et al., 2007) over (Xing et al., 2002) is that it is computationally more efficient.

(Klein et al., 2002) has a favourable property we want to keep. The constraints are imposed strictly, meaning that no cannot-linked instances can appear in the same cluster. I.e., they are hard constraints. In the case of metric learning, the constraints are not imposed strictly. In a new learned distance metric, two cannot-linked instances will be relatively distant, but there is no guarantee they will not end up in the same cluster. Although we think that metric learning makes a better use of provided constraints, we do not want to lose the benefit of hard constraints. Thus, we use both mechanisms in combination to impose constraints. We first learn a Mahalanobis distance based on the provided constraints. Then, we compute distance matrix and employ the mechanism proposed by (Klein et al., 2002) on the learned distance matrix.

4.2.2 Active Constraint Generation

As mentioned before, the choice of the set of constraints affects the quality of the end clustering. In this work, we define a novel method to choose informative instances, which we believe will have maximum impact on the end cluster quality, when they are labeled and used to generate constraints for our task. We use an iterative approach. Each iteration consists of three steps: (1) generating clusters by the process described in Section 4.2.1 imposing available constraints, (2) choosing the most informative instance, considering the cluster boundaries, and acquiring its label, (3) extending the available constraints with the ones we generate from the newly labeled instance.

We consider an instance to be informative if there is a high probability that the knowledge of its label may change the cluster boundaries. The more probable that change is, the more informative is the instance. The basic idea is that if an instance is in a cluster holding instances of type a and it is close to another cluster holding instances of type b, that instance is most likely misclustered. Thus, it should be queried. Our hypothesis is that, in each iteration, the algorithm will choose the most problematic – informative – instance that will end up changing cluster boundaries. This will result in each iteration in a more reliable distance metric, which in return will provide more reliable estimates of problematic instances in future iterations. The imposed con-

Algorithm 1 Iterative Constrained Clustering

```
1:  C = $cluster$(I)
2:  $I_{\{L\}}$ = $labelprototypes$(C)
3:  **while** $|I_{\{L\}}| <$ stop **do**
4:      Con = $createconstraints(I_{\{L\}})$
5:      $Matrix_{dist}$ = $learnmetric$(I, Con)
6:      C = $constraintedcluster(Matrix_{dist}, Con)$
7:      L = $labelmostinformative$(C)
8:      $I_{\{L\}} = I_{\{L\}} \cup L$
9:  **end while**
10: $propagatelabels(I_{\{L\}}, C)$    {C...Clusters; Con...Constraints;
    I...Instances; $I_{\{L\}}$...Labeled Instances; $Matrix_{dist}$...Distance Matrix}
```

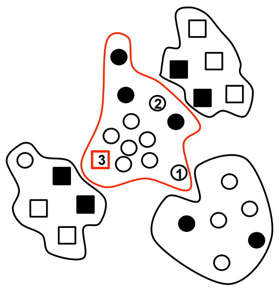

Figure 1: Behaviour of selection function

straints will move the clustering in each iteration towards better separation of S and O instances.

To define informativeness, we define a scoring function, which is used to score each data point on its goodness. The lower the score, the more likely it is that the instance is mis-clustered. Choosing the data point with the lowest score will likely change clustering borders in the next iteration. Our scoring function is based on the *silhouette coefficient*, a popular unsupervised cluster validation metric to measure goodness (Tan et al., 2005) of a cluster member. Basically, the silhouette score assigns a cluster member that is close to another cluster a lower score, and a cluster member that is closer to the cluster center a higher score. That is partly what we want. In addition, we do not want to penalize a cluster member that is close to another cluster having members with the same label. For this purpose, we calculate the silhouette score only over clusters with an opposing label (i.e., holding members with an opposing label). In addition, we consider only instances labeled so far when computing the score. We call this new coefficient silh$_{const}$. It is computed as follows: (1) for an instance i, compute its average distance from the other instances in its cluster x_i which are already labeled, (2) for an instance i, compute its average distance from the labeled instances of the clusters from an opposing label and take the minimum of these averages y_i, (3) compute the silhouette coefficient as $(y_i - x_i) / \max(y_i, x_i)$.

The silh$_{const}$ coefficient has favourable properties. First, it scores members that are close to a cluster with an opposing label lower than the members that are close to a cluster with the same label. According to our definition, these members are more informative. Figure 1 holds a sample cluster setting. The shape of a member denotes its label and its fill denotes whether or not it has been queried. In this example, silh$_{const}$ scores members 2 and 3 lower than 1. Thus, member 1 will not be selected, which is the right decision in this example. Both members 2 and 3 are close to clusters with an opposing label. In this example silh$_{const}$ scores member 3 lower, which is farther away from already labeled members in the cluster. Thus, member 3 will be selected to be labeled. This type of behaviour results in an explorative strategy.

The active selection strategy proposed by (Klein et al., 2002) is single pass. Thus, it does not have the opportunity to observe the complete cluster structure before choosing constraints. We hypothesize that our strategy will provide more informative constraints, since it has the advantage of being able to base the decision of which constraints to generate on fully observed cluster structure in each iteration.

We call our proposed algorithm *Iterative Constrained Clustering (ICC)*. In our final implementation, ICC starts by simply clustering the instances without any constraints. The algorithm queries the label of the prototypical member – the member closest to the cluster center – of each cluster. Then, the described iterations begin. Algorithm 1 contains the complete ICC algorithm. Note that line 6 is equivalent to the algorithm of (Klein et al., 2002).

4.3 Experiments

This section gives details on experiments to evaluate the purity of the semi-automatically generated subjectivity sense tagged data by our "cluster and label" strategy. We carry out detailed analysis to quantify the effect of the proposed active selection strategy and of metric learning on the purity of the generated data. We compare our active selection strategy to random selection and also to (Klein et al., 2002). The comparison is done on the senSWSD dataset. SenSWSD consists of three subsets, SENSEVAL I,II and III. Since we devel-

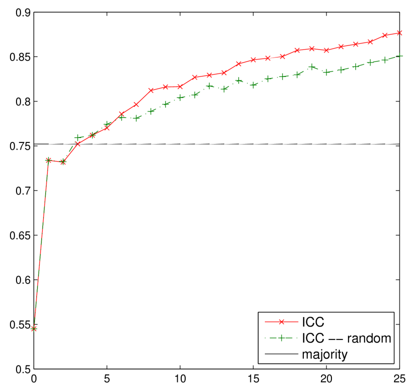

Figure 2: Label Purity – ICC vs. random selection

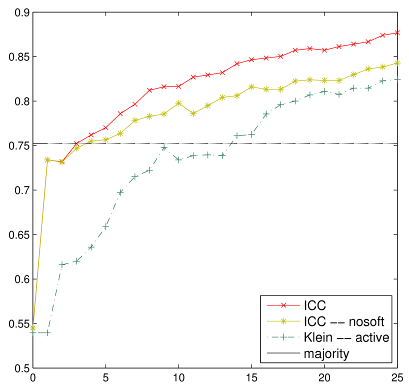

Figure 3: Label Purity – ICC vs. Klein

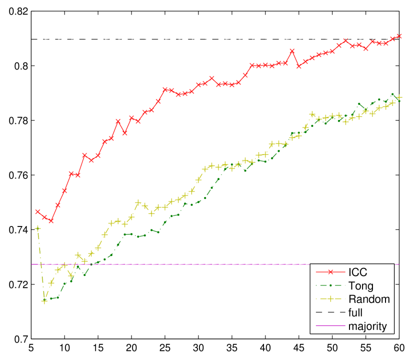

Figure 4: SWSD accuracy on ICC generated data

oped our active selection algorithm on the SENSEVAL I subset, we use only SENSEVAL II and III subsets for comparison. We apply ICC to each word in the comparison set separately, and report cumulative results for the purity of the generated data. We report results for different percentages of the queried data amount (e.g. 10% means that the algorithm queried 10% of the data to create constraints). This way, we obtain a learning curve. We fix the cluster number to 7 as in the context representation experiments.

4.3.1 Effect of Active Selection Strategy

Figure 2 holds the comparison of ICC with silh$_{const}$ selection to a random selection baseline. "majority" stands for majority label frequency in the dateset. We see that silh$_{const}$ performs better than the random selection. By providing labels to only 25% of the data, we can achieve 87.67% pure fully labeled data.

For comparison, we also evaluate the performance of (Klein et al., 2002) with their active constraint selection strategy as described in Section 4.1. Note that originally (Klein et al., 2002) requests the constraint between two roots. In our setting, it requests labels of the roots and then generates constraints from the obtained labels. Since we have a binary task, querying labels makes more sense. This has the advantage that more constraints from each request are obtained. Moreover, it allows a direct comparison to our algorithm. (Klein et al., 2002) does not use any metric learning. Thus, we run our algorithm also without metric learning, in order to compare the effectiveness of both active selection strategies fairly. In Figure 3, we see that silh$_{const}$ performs better than the active selection strategy described in (Klein et al., 2002). We also see that metric learning results in a big improvement. In addition, metric learning results in a smoother learning curve, which is a favourable property for a real-world application.

4.3.2 SWSD on semi-automatically generated annotations

Now that we have a tool to generate training data for SWSD, we want to evaluate it on the actual SWSD task. We want to see if the obtained purity is enough to create reliable SWSD classifiers. For this purpose, we test ICC on mturkSWSD dataset.

For each word in our dataset, we conduct 10-fold cross-validation experiments. ICC is applied to training folds to label instances semi-automatically. We train SWSD classifiers on the generated training fold labels and test the classifiers on the corresponding test fold. We distinguish between queried instances and propagated labels. The queried instances are weighted as 1 and the instances with propagated labels are weighted by their silh$_{const}$ score, since that measure gives the goodness of an instance. The score is defined between -1 and 1. This score is normalized between 0 and 1, before it is used as a weight. SVM classifiers from the Weka package (Witten and Frank., 2005) with its default settings are used as in (Akkaya et al., 2011).

We implement two baselines. The first is simple *random sampling* and the second is *uncertainty sampling*, which is an *active learning* (AL) method. We use "simple margin" selection as described in (Tong and Koller, 2001). It selects, in each iteration, the instance closest to the decision boundary of the trained SVM. Each method is run until it reaches the accuracy of training fully on the gold-standard data. ICC reaches that boundary when provided only 59% of the labels in the dataset. For uncertainty sampling and random sampling, these values are 92% and 100%, respectively. In Figure 4, we see the SWSD accuracy for different queried data percentages. "full" stands for training fully on gold-standard data. We see that training SWSD on semi-automatically labeled data by ICC does consistently better than uncertainty sampling and random sampling.

It is surprising to see that uncertainty sampling overall does not do better than random sampling. We believe that it might be because of sampling bias. During AL, as more and more labels are obtained, the training set quickly diverges from the underlying data distribution. (Schütze et al., 2006) states that AL can explore the feature space in such a biased way that it can end up ignoring entire clusters of unlabeled instances. We think that SWSD is highly prone for the mentioned missed cluster problem because of its unique nature. As mentioned, SWSD is a binary task where we distinguish between subjective and objective usages of a subjectivity word. Although the classification is binary, the underlying usages are grouped into multiple clusters corresponding to senses of the word. It is possible that two groups of usages which are represented quite differently in the feature space are both subjective or objective. Moreover, one usage group might be closer to a usage group from the opposing label than to a group with the same label.

We see that our method reduces the annotation amount by 36% in comparison to uncertainty sampling and by 41% in comparison to random sampling to reach the performance of the SWSD system trained on fully annotated data.

5 Related Work

One related line of research is constrained clustering also known as semi-supervised clustering (Xing et al., 2002; Wagstaff and Cardie, 2000; Grira et al., 2004; Demiriz et al., 1999). It has been applied to various datasets and tasks such as image and document categorization. To our knowledge, we are the first to utilize constrained clustering for a difficult NLP task.

There have been only two previous works selecting constraints for constrained clustering actively (Basu et al., 2004; Klein et al., 2002). The biggest difference of our approach is that it is iterative as opposed to single pass.

Active Learning (AL) (Settles, 2009; Settles and Craven, 2008; Hwa, 2004; Tong and Koller, 2001) builds another important set of related work. Our method is inspired by uncertainty sampling. We accomplish active selection in the clustering setting.

6 Conclusions

In this paper, we explore a "cluster and label" strategy to reduce the human annotation effort needed to generate subjectivity sense-tagged data. In order to keep the noise in the semi-automatically labeled data minimal, we investigate different feature space types and evaluate their expressiveness. More importantly, we define a new algorithm called iterative constrained clustering (ICC) with an active constraint selection strategy. We show that we can obtain a fairly reliable labeled data when we utilize ICC.

We show that the active selection strategy we propose outperforms a previous approach by (Klein et al., 2002) for generating subjectivity sense-tagged data. Training SWSD classifiers on ICC generated data improves over random sampling and uncertainty sampling (Tong and Koller, 2001). We can achieve on mturkSWSD 36% annotation reduction over uncertainty sampling and 41% annotation reduction over random sampling in order to reach the performance of SWSD classifiers trained on fully annotated data.

To our knowledge, this work is the first application of constrained clustering to a hard NLP problem. We showcase the power of constrained clustering. We hope that the same "cluster and label" strategy will be applicable to Word Sense Disambiguation. This will be part of our future work.

7 Acknowledgments

This material is based in part upon work supported by National Science Foundation awards #0917170 and #0916046.

References

Apoorv Agarwal, Fadi Biadsy, and Kathleen Mckeown. 2009. Contextual phrase-level polarity analysis using lexical affect scoring and syntactic N-grams. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 24–32, Athens, Greece, March. Association for Computational Linguistics.

Cem Akkaya, Janyce Wiebe, and Rada Mihalcea. 2009. Subjectivity word sense disambiguation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 190–199, Singapore, August. Association for Computational Linguistics.

Cem Akkaya, Janyce Wiebe, Alexander Conrad, and Rada Mihalcea. 2011. Improving the impact of subjectivity word sense disambiguation on contextual opinion analysis. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 87–96, Portland, Oregon, USA, June. Association for Computational Linguistics.

Cem Akkaya, Janyce Wiebe, and Rada Mihalcea. 2012. Utilizing semantic composition in distributional semantic models for word sense discrimination and word sense disambiguation. In *ICSC*, pages 45–51.

Alina Andreevskaia and Sabine Bergler. 2008. When specialists and generalists work together: Overcoming domain dependence in sentiment tagging. In *Proceedings of ACL-08: HLT*, pages 290–298, Columbus, Ohio, June. Association for Computational Linguistics.

Sugato Basu, Arindam Banerjee, and R. Mooney. 2002. Semi-supervised clustering by seeding. In *In Proceedings of 19th International Conference on Machine Learning (ICML-2002)*.

Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. 2004. Active semi-supervision for pairwise constrained clustering. In *SDM*.

Kenneth Bloom, Navendu Garg, and Shlomo Argamon. 2007. Extracting appraisal expressions. In *HLT-NAACL 2007*, pages 308–315, Rochester, NY.

John Bullinaria and Joseph Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39:510–526. 10.3758/BF03193020.

Ian Davidson, Kiri Wagstaff, and Sugato Basu. 2006. Measuring constraint-set utility for partitional clustering algorithms. In *PKDD*, pages 115–126.

Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. 2007. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 209–216, New York, NY, USA. ACM.

Ayhan Demiriz, Kristin Bennett, and Mark J. Embrechts. 1999. Semi-supervised clustering using genetic algorithms. In *In Artificial Neural Networks in Engineering (ANNIE-99*, pages 809–814. ASME Press.

Nizar Grira, Michel Crucianu, and Nozha Boujemaa. 2004. Unsupervised and semi-supervised clustering: a brief survey. In *in A Review of Machine Learning Techniques for Processing Multimedia Content, Report of the MUSCLE European Network of Excellence*.

Rebecca Hwa. 2004. Sample selection for statistical parsing. *Comput. Linguist.*, 30(3):253–276, September.

Adam Kilgarriff. 1997. I dont believe in word senses. *Computers and the Humanities*, 31(2):91–113.

Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *Proceedings of the Twentieth International Conference on Computational Linguistics (COLING 2004)*, pages 1267–1373, Geneva, Switzerland.

D. Klein, K. Toutanova, I.T. Ilhan, S.D. Kamvar, and C. Manning. 2002. Combining heterogeneous classifiers for word-sense disambiguation. In *Proceedings of the ACL Workshop on "Word Sense Disambiguatuion: Recent Successes and Future Directions*, pages 74–80, July.

R. Mihalcea. 2002. Instance based learning with automatic feature selection applied to Word Sense Disambiguation. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, Taipei, Taiwan, August.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

A. Purandare and T. Pedersen. 2004. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL 2004)*, Boston.

Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 105–112, Sapporo, Japan.

Magnus Sahlgren. 2006. *The Word-Space Model: using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces.* Ph.D. thesis, Stockholm University.

Hinrich Schütze, Emre Velipasaoglu, and Jan O. Pedersen. 2006. Performance thresholding in practical text classification. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, CIKM '06, pages 662–671, New York, NY, USA. ACM.

H. Schutze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1070–1079, Stroudsburg, PA, USA. Association for Computational Linguistics.

Burr Settles. 2009. Active Learning Literature Survey. Technical Report 1648, University of Wisconsin–Madison.

Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. 2005. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Simon Tong and Daphne Koller. 2001. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. March.

Peter Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, pages 417–424, Philadelphia, Pennsylvania.

Kiri Wagstaff and Claire Cardie. 2000. Clustering with instance-level constraints. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML-2000)*, pages 1103–1110.

Casey Whitelaw, Navendu Garg, and Shlomo Argamon. 2005. Using appraisal taxonomies for sentiment analysis. In *Proceedings of CIKM-05, the ACM SIGIR Conference on Information and Knowledge Management*, Bremen, DE.

I. Witten and E. Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition*. Morgan Kaufmann, June.

Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart J. Russell. 2002. Distance metric learning with application to clustering with side-information. In *NIPS*, pages 505–512.

Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 129–136, Sapporo, Japan.