Noise or additional information? Leveraging crowdsource annotation item agreement for natural language tasks.

Emily K. Jamison ‡ and Iryna Gurevych †‡
‡Ubiquitous Knowledge Processing Lab (UKP-TUDA), Department of Computer Science, Technische Universität Darmstadt
† Ubiquitous Knowledge Processing Lab (UKP-DIPF), German Institute for Educational Research
http://www.ukp.tu-darmstadt.de

Abstract

In order to reduce noise in training data, most natural language crowdsourcing annotation tasks gather redundant labels and aggregate them into an integrated label, which is provided to the classifier. However, aggregation discards potentially useful information from linguistically ambiguous instances.

For five natural language tasks, we pass item agreement on to the task classifier via soft labeling and low-agreement filtering of the training dataset. We find a statistically significant benefit from low item agreement training filtering in four of our five tasks, and no systematic benefit from soft labeling.

1 Introduction

Crowdsourcing is a cheap and increasingly-utilized source of annotation labels. In a typical annotation task, five or ten labels are collected for an instance, and are aggregated together into an integrated label. The high number of labels is used to compensate for worker bias, task misunderstanding, lack of interest, incompetence, and malicious intent (Wauthier and Jordan, 2011).

Majority voting for label aggregation has been found effective in filtering noisy labels (Nowak and Rüger, 2010). Labels can be aggregated under weighted conditions reflecting the reliability of the annotator (Whitehill et al., 2009; Welinder et al., 2010). Certain classifiers are also robust to random (unbiased) label noise (Tibshirani and Manning, 2014; Beigman and Beigman Klebanov, 2009). However, minority label information is discarded by aggregation, and when the labels were gathered under controlled circumstances, these labels may reflect linguistic intuition and contain useful information (Plank et al., 2014b). Two alternative strategies that allow the classifier to learn from the item agreement include training instance filtering and soft labeling. Filtering training instances by item agreement removes low agreement instances from the training set. Soft labeling assigns a classifier weight to a training instance based on the item agreement.

Consider two Affect Recognition instances and their Krippendorff (1970)'s $\alpha$ item agreement:

**Text:** India’s Taj Mahal gets facelift

Sadness Rating (0-100): 8.0

$\alpha$ Agreement (-1.0 – 1.0): 0.7

Figure 1: Affect Recognition Easy Case.

**Text:** After Iraq trip, Clinton proposes war limits

Sadness Rating (0-100): 12.5

$\alpha$ Agreement (-1.0 – 1.0): -0.1

Figure 2: Affect Recognition Hard Case.

In Figure 1, annotators mostly agreed that the headline expresses little sadness. But in Figure 2, the low item agreement may be caused by instance difficulty (i.e., *Is a war zone sad or just bad?*): a Hard Case (Zeman, 2010). Previous work (Beigman Klebanov and Beigman, 2014; Beigman and Beigman Klebanov, 2009) has shown that training strategy may affect Hard and Easy Case test instances differently.

In this work, for five natural language tasks, we examine the impact of passing crowdsource item agreement on to the task classifier, by means of training instance filtering and soft labeling. We construct classifiers for Biased Text Detection, Stemming Classification, Recognizing Textual Entailment, Twitter POS Tagging, and Affect Recognition, and evaluate the effect of our different training strategies on the accuracy of each
task. These tasks represent a wide range of machine learning tasks typical in NLP: sentence-level SVM regression using n-grams; word pairs with character-based features and binary SVM classification; pairwise sentence binary SVM classification with similarity score features; CRF sequence word classification with a range of feature types; and sentence-level regression using a token-weight averaging, respectively. We use pre-existing, freely-available crowdsourced datasets and post all our experiment code on GitHub\(^1\).

**Contributions** This is the first work (1) to apply item-agreement-weighted soft labeling from crowdsourced labels to multiple real natural language tasks; (2) to filter training instances by item agreement from crowdsourced labels, for multiple natural language tasks; (3) to evaluate classifier performance on high item agreement (Easy Case) instances and low item agreement (Hard Case) instances across multiple natural language tasks.

2 Related Work

Dekel and Shamir (2009) calculated integrated labels for an information retrieval crowdsourced dataset, and identified low-quality workers by deviation from the integrated label. Removal of these workers’ labels improved classifier performance on data that was not similarly filtered. While much work (Dawid and Skene, 1979; Ipeirotis et al., 2010; Dalvi et al., 2013) has explored techniques to model worker ability, bias, and instance difficulty while aggregating labels, there is no evaluation comparing classifiers trained on the new integrated labels with other options, on their respective NLP tasks.

Training instance filtering aims to remove mislabeled instances from the training dataset. Sculley and Cormack (2008) learned a logistic regression classifier to identify and filter noisy labels in a spam email filtering task. They also proposed a label correcting technique that replaces identified noisy labels with “corrected” labels, at the risk of introducing noise into the corpus. Rebbapragada et al. (2009) developed a label noise detection technique to cluster training instances and remove label outliers. Raykar et al. (2010) jointly learned a classifier/regressor, annotator accuracy, and the integrated label on datasets with multiple noisy labels, outperforming Smyth et al. (1995)’s model of estimating ground truth labels.

Soft labeling, or the association of one training instance with multiple, weighted, conflicting labels, is a technique to model noisy training data. Thiel (2008) found that soft labeled training data produced more accurate classifiers than hard labeled training data, with both Radial Basis Function Networks and Fuzzy-Input Fuzzy-Output SVMs. Shen and Lapata (2007) used soft labeling to model their semantic frame structures in a question answering task, to represent that the semantic frames can bear multiple semantic roles.

Previous research has found that, for a few individual NLP tasks, training while incorporating label noise weight may produce a better model. Martínez Alonso et al. (2015) show that informing a parser of annotator disagreement via loss function reduced error in labeled attachments by 6.4%. Plank et al. (2014a) incorporate annotator disagreement in POS tags into the loss function of a POS-tag machine learner, resulting in improved performance on downstream chunking.

Beigman Klebanov and Beigman (2014) observed that, on a task classifying text as semantically old or new, the inclusion of Hard Cases in training data resulted in reduced classifier performance on Easy Cases.

3 Overview of Experiments

We built systems for the five NLP tasks, and trained them using aggregation, soft labeling, and instance screening strategies. When labels were numeric, the integrated label was the average\(^2\). When labels were nominal, the integrated label was majority vote. Krippendorff (1970)’s \(\alpha\) item agreement was used to filter ambiguous training instances. For soft labeling, percentage item agreement was used to assign instance weights.

We followed Sheng et al. (2008)’s suggested *Multiplied Examples* procedure: for each unlabeled instance \(x_i\) and each existing label \(y_{ij} \in L_i = \{y_{ij}\}\) (as annotated by worker \(j\)), we create one replica of \(x_i\), assign it \(y_{ij}\), and weight the instance according to the count of \(y_{ij}\) in \(L_i\) (i.e., the percentage item agreement). For each training strategy (Soft-Label, etc.), the training instances were changed by the strategy, but the test instances were unaffected. For the division of test instances into Hard

\(^1\)github.com/EmilyKJamison/crowdsourcing

\(^2\)We followed Yano et al. (2010) and Strapparava and Mihalcea (2007) in using mean as gold standard. Although another aggregation such as as median might be more representative, such discussion is beyond the scope of this paper.
and Easy Cases, the training instances were unaffected, but the test instances were filtered by \( \alpha \) item agreement. Hard/Easy Case parameters were chosen to divide the corpus by item agreement into roughly equal portions\(^3\), relative to the corpus, for post-hoc error analysis.

All systems except Affect Recognition were constructed using DKPro Text Classification (Daxenberger et al., 2014), and used Weka’s SMO (Platt, 1999) or SMOreg (Shevade et al., 2000) implementations with default parameters, with 10-fold (or 5-fold, for computationally-intensive POS Tagging) cross-validation. More details are available in the Supplemental Notes document.

**Agreement Parameters** Training strategies HighAgree and VeryHigh utilize agreement cutoff parameters that vary per corpus. These strategies are a discretized approximation of the gradual effect of filtering low agreement instances from the training data. For any given corpus, we could not use a cutoff value equal to no filtering, or that eliminated a class. If there were only 2 remaining cutoffs, we used these. If there were more candidate cutoff values, we trained and evaluated a classifier on a development set and chose the value for HighAgree that maximized Hard Case performance on the development set.

**Percentage Agreement** In this paper, we follow Beigman Klebanov and Beigman (2014) in using the nominal agreement categories Hard Cases and Easy Cases to separate instances by item agreement. However, unlike Beigman Klebanov and Beigman (2014) who use simple percentage agreement, we calculate item-specific agreement via Krippendorff (1970)’s \( \alpha \) item agreement\(^4\), with Nominal, Ordinal, or Ratio distance metrics as appropriate. The agreement is expressed in the range (\(-1.0 - 1.0\); 1.0 is perfect agreement).

### 3.1 Biased Language Detection

This task detects the use of bias in political text. The corpus (Yano et al., 2010)\(^5\) consists of 1,041 sentences from American political blogs. For each sentence, five crowdsource annotators chose a label no bias, some bias, and very biased. We follow Yano et al. (2010) in representing the amount of bias on a numerical scale (1-3). Hard/Easy Case cutoffs were <-.21 and >.20. Of 1041 total instances, 161 were Hard Cases (<-.21) and 499 were Easy Cases (>-.20).

We built an SVM regression task using unigrams, to predict the numerical amount of bias. The gold standard was the integrated labels. Item-specific agreement was calculated with Ordinal Distance Function (Krippendorff, 1980).

We used the following training strategies: VeryHigh Filtered for agreement >.0.4. HighAgree Filtered for agreement >-.2. SoftLabel One training instance is generated for each label from a text, and weighted by how many times that label occurred with the text. SLLimited SoftLabel, except that training instances with a label distance >1.0 from the original text label average are discarded.

### 3.2 Morphological Stemming

The goal of this binary classification task is to predict, given an original word and a stemmed version of the word, whether the stemmed version has been correctly stemmed. The word pair was correct if: the stemmed word contained one less affix; or if the original word was a compound, the stemmed word had a space inserted between the components; or if the original word was misspelled, the stemmed word was deleted; or if the original word had no affixes and was not a compound and was not misspelled, then the stemmed word had no changes.

This dataset was compiled by Carpenter et al. (2009)\(^6\). The dataset contains 6679 word pairs; most pairs have 5 labels each. In the cross-validation division, no pairs with the same original word could be split across training and test data. The gold standard was the integrated label, with 4898 positive and 1781 negative pairs. Hard/Easy Case cutoffs were <-.5 and >.5. Of 6679 total instances, 822 were Hard Cases (<-.5) and 3615 were Easy Cases (>-.5). Features used are combinations of the characters after the removal of the longest common substring between the word pair, including 0-2 additional characters from the substring: word boundaries are marked.

Stemming-new training strategies include: HighAgree Filtered for agreement >-.1. SLLimited MajVote with instances weighted by the frequency of the label for the text pair.

---

\(^3\)Limited by the discrete nature of our agreement.

\(^4\)From the DKPro Statistics library (Meyer et al., 2014)

\(^5\)Available at sites.google.com/site/amtworkshop2010/data-1

\(^6\)Available at github.com/bob-carpenter/anno
3.3 Recognising Textual Entailment

Recognizing textual entailment is the process of determining if, given two sentences text and hypothesis, the meaning of the hypothesis can be inferred from the text.

We used the dataset from the PASCAL RTE-1, which contains 800 sentence pairs. The crowdsource annotations of 10 labels per pair were obtained by Snow et al. (2008)\(^7\). We reproduced the basic system described in (Dagan et al., 2006) of TF-IDF weighted Cosine Similarity between lemmas of the text and hypothesis. The weight of each word \(w \in \text{document} \), \(n \) total documents, is the log-plus-one term frequency normalized by raw term \(n \) document frequency, with Euclidean normalization.

\[
\text{weight}(i, j) = \begin{cases} 
(1 + \log(tf_{i,j}))^{\frac{N}{\text{term}}} & \text{if } tf_{i,j} \geq 1 \\
0 & \text{if } tf_{i,j} = 0 
\end{cases}
\]

Additionally, we used features including the difference in noun chunk character and token length, the difference in number of tokens, shared named entities, and subtask names. The gold standard was the original labels from Dagan et al. (2006). Hard/Easy Case cutoffs were <0.0 and >.3. Training strategies are from Biased Language (VeryHigh) and Stem (others) experiments, except the HighAgree cutoff was 0.0 and the VeryHigh cutoff was 0.3. Of 800 total instances, 230 were Hard Cases (<0.0) and 207 were Easy Cases (>0.3).

3.4 POS tagging

We built a POS-tagger for Twitter posts. We used the training section of the dataset from Gimpel et al. (2011). The POS tagset was the universal tagset (Petrov et al., 2012); we converted Gimpel et al. (2011)’s tags to the universal tagset using Hovy et al. (2014)’s mapping. Crowdsource labels for this data came from Hovy et al. (2014)\(^8\), who obtained 5 labels for each tweet. After aligning and cleaning, our dataset consisted of 953 tweets of 14,439 tokens.

We followed Hovy et al. (2014) in constructing a CRF classifier (Lafferty et al., 2001), using a list of English affixes, Hovy et al. (2014)’s set of orthographic features, and word clusters (Owoputi et al., 2013). In the cross-validation division, individual tweets were assigned to folds. The gold standard was the integrated label. Hard/Easy Case cutoffs were <0.0 and >.49. Of 14,439 tokens, 649 were Hard Cases (<0.0) and 10830 were Easy Cases (>0.49).

We used the following strategies:

**VeryHigh** For each token \(t \) in sequence \(s \) where \(\text{agreement}(t) < 0.5 \), \(s \) is broken into two separate sequences \(s_1 \) and \(s_2 \) and \(t \) is deleted (i.e. filtered).

**HighAgree** VeryHigh with agreement <0.2.

**SoftLabel** For each proto-sequence \(s \), we generate 5 sequences \(\{s_0, s_1, ..., s_5\} \), in which each token \(t \) is assigned a crowdsource label drawn at random: \(l_{t,s_i} \in L_t \).

**SLLimited.** Each token \(t \) in sequence \(s \) is assigned its MajVote label. Then \(s \) is given a weight representing the average item agreement for all \(t \in s \).

3.5 Affect Recognition

Our Affect Recognition experiments are based on the affective text annotation task in Strapparava and Mihalcea (2007), using the Sadness dataset. Each headline is rated for “sadness” using a scale of 0-100. Examples are in Figures 1 and 2. We use the crowdsourced annotation for a 100-headline sample of this dataset provided by Snow et al. (2008)\(^9\), with 10 annotations per emotion per headline. Of 100 total instances, 20 were Hard Cases (<0.0) and 49 were Easy Cases (>0.30).

Our system design is identical to Snow et al. (2008), which is similar to the SWAT system (Katz et al., 2007), a top-performing system on the SemEval Affective Text task. Hard/Easy Case cutoffs were <0.0 and >.3.

Training strategies are the same as for the Biased Language experiments, except:

**VeryHigh** Filtered for agreement >0.3.

**HighAgree** Filtered for agreement >0.

**SLLimited** SoftLabel, except that instances with a label distance >20.0 from the original label average are discarded.

4 Results

Our results on all five tasks, using each of the training strategies and variously evaluating on all, Easy, or Hard Cases, can be seen in Table 1. Systems outputting numeric values show results in Pearson correlation, and systems outputting nominal labels show micro F\(_1\). Soft labeling (SoftLabel) failed to outperform integrated labels for 4 of the 5 complete test sets. Likewise, **SLLimited**
did not significantly outperform Integrated. However, HighAgree does outperform Integrated on 4 or the 5 tasks, especially for Hard Cases: Hard Case improvements for Biased Language and POS Tagging, and Affective Text, and overall improvements for RTE, POS Tagging, and Affective Text were significant (Paired TTest, \( p < 0.05 \), for numerical output, or McNemar’s Test\(^{10} \) (McNemar, 1947), \( p < 0.05 \), for nominal classes). The fifth task, Stemming, had the lowest number of item agreement categories of the five tasks, preventing fine-grained agreement training filtering, which explains why filtering shows no benefit.

All training strategies used the same amount of annotated data as input, and for filtering strategies such as HighAgree, a reduced number of strategy-output instances are used to train the model. As a higher cutoff is used for HighAgree, the lack of training data results in a worse model; this can be seen in the downward curves of Figures 3 – 6, where the curved line is HighAgree and the matching pattern straight line is Integrated. (Due to the low number of item agreement categories, Stemming results are not displayed in an item agreement cutoff table.) However, Figures 4 – 6 show the overall performance boost, and Figure 3 shows the Hard Case performance boost, right before the downward curves from too little training data, using HighAgree.

Comparability We found the accuracy of our systems was similar to that reported in previous literature. Dagan et al. (2006) report performance of the RTE system, on a different data division, with accuracy=0.568. Hovy et al. (2014) report majority vote results (from acc=0.805 to acc=0.837 on a different data section) similar to our results of 0.790 micro-F\(_1\). For Affective Text, Snow et al. (2008) report results on a different data section of \( r=0.174 \), a merged result from systems trained on combinations of crowdsourced labels and evaluated against expert-trained systems. The SWAT system (Katz et al., 2007), which also used lexical resources and additional training data, achieved \( r=0.3898 \) on a different section of data. These results are comparable with ours, which range from \( r=0.326 \) to \( r=0.453 \).

5 Conclusions and Future Work

In this work, for five natural language tasks, we have examined the impact of informing the classifier of crowdsourced item agreement, by means of soft labeling and removal of low-agreement training instances. We found a statistically significant benefit from low-agreement training filtering in four of our five tasks, and strongest improvements for Hard Cases. Previous work (Beigman Kleinov and Beigman, 2014) found a similar effect, but only evaluated a single task, so generalizability was unknown. We also found that soft labeling was not beneficial compared to aggregation. Our findings suggest that the best crowdsourced label training strategy is to remove low item agreement instances from the training set.

Acknowledgments

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806, and by the Center for Advanced Security Research (www.cased.de).

---

\(^{10}\)See Japkowicz and Shah (2011) for usage description.
References


Olatobi Owoputi, Brendan O’Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for


