xiacui at SemEval-2023 Task 11: Learning a Model in Mixed-Annotator Datasets using Annotator Ranking Scores as Training Weights

Xia Cui
Manchester Metropolitan University
x.cui@mmu.ac.uk

Abstract

This paper describes the development of a system for SemEval-2023 Shared Task 11 on Learning with Disagreements (Le-Wi-Di) (Leonardellli et al., 2023). Labelled data plays a vital role in the development of machine learning systems. The human-annotated labels are usually considered the *truth* for training or validation. To obtain *truth* labels, a traditional way is to hire domain experts to perform an expensive annotation process. Crowd-sourcing labelling is comparably cheap, whereas it raises a question on the reliability of annotators. A common strategy in a mixed-annotator dataset with various sets of annotators for each instance is to aggregate the labels among multiple groups of annotators to obtain the *truth* labels. However, these annotators might not reach an agreement, and there is no guarantee of the reliability of these labels either. With further problems caused by human label variation, subjective tasks usually suffer from the different opinions provided by the annotators. In this paper, we propose two simple heuristic functions to compute the annotator ranking scores, namely *AnnoHard* and *AnnoSoft*, based on the *hard labels* (i.e., aggregative labels) and *soft labels* (i.e., cross-entropy values). By introducing these scores, we adjust the weights of the training instances to improve the learning with disagreements among the annotators.

1 Introduction

Annotated datasets are fundamental for any machine learning model. Traditional supervised machine learning models are heavily based on well-labelled datasets. Fine-tuning and validation could not exist without them either. The cost of manual labelling using domain experts is generally high to ensure the annotation quality. With the recent popularity of crowd-sourcing, the cost has become lower. However, the reliability of annotators remains a question. Even with domain experts, disagreements are commonly observed in the multiple annotations for the same task due to several circumstances. For example, the task is complex with many factors. It may require annotators' expertise, such as predicting mental health issues (Thieme et al., 2020), or subjective such as detecting emotions or other polarised opinions (Schuff et al., 2017; Akhtar et al., 2020). The model trained with unreliable labels can potentially increase the cost by human post-validation.

In this paper, we focus on a specific sub-problem of learning with disagreements in mixed sets of annotators from different backgrounds, and it could be raised by crowd-sourcing or a mixture of the crowd and expert labels. We propose a training strategy to weigh the instance by computing the ranking score of the annotator. More specifically, our method is proposed to trust more instances with higher annotator ranks during the training stage. Hence, we propose two functions to compute annotator ranking scores: *AnnoHard* and *AnnoSoft*. As suggested in the naming, one is computed using the hard labels, and the other uses the soft labels. This paper follows the task definition to consider cross entropy values as the *soft labels* and the majority voting labels as the *hard labels*.

By participating in this shared task, we made an initial attempt using heuristic methods and explored the performance of introducing *AnnoHard* and *AnnoSoft* to various learning algorithms. A list of main contributions is summarised as follows:

- We proposed and applied two functions of annotator ranks (i.e., *AnnoHard* and *AnnoSoft*) to four binary classification tasks from the Le-Wi-Di shared task. We conducted the experiments using seven learning algorithms and eight document representation methods.
- By introducing the annotator ranking scores, the trained model could capture more patterns from reliable training instances and slightly improve both soft and hard evaluation.
- We found that using TF-IDF with Random

Proceedings of the The 17th International Workshop on Semantic Evaluation (SemEval-2023), pages 1076–1084
July 13-14, 2023 ©2023 Association for Computational Linguistics

Forest shows the best performance on the three short text datasets, regardless of the language used in the text and whether it coped with the annotator ranking scores.

The source code for this paper is publicly available on GitHub[1].

2 Background

In this section, we specifically focus on the prior works driven by the dependent scores of annotators. As the annotator is the origin of the disagreements, these works have demonstrated the design of training strategies to address this factor. Subjective tasks (e.g., detecting hate speech or offensive languages) generally have different definitions among different communities (Akhtar et al., 2020; Poletto et al., 2021). Developing an automatic system based on the hard labels would substantially impact a specific community. To overcome this issue, Akhtar et al. (2020) proposed grouping the annotators into two groups using the average Polarization index (P-index). However, polarity is a task-specific factor. Plank et al. (2014) proposed computing the two annotators' agreements using F1 scores between them and label confusion probabilities for Part-of-Speech (POS) tagging. The method was extended to dependency parsing in Alonso et al. (2015). We aim to propose a method to increase the feature and task generality to the dedicated problem of disagreements.

Machine learning models are trained by taking the penalties for misclassifications into account. A loss function is applied to map the distance between the current output and the ground truth. Each training instance could possibly affect the prediction result (Plank et al., 2014). Introducing instance weights to cost-sensitive classifiers is a popular solution. It intends to increase or decrease the weights of some instances in NLP tasks to enrich the model performance (Geibel and Wysotzki, 2003; Higashiyama et al., 2013; Plank et al., 2014; Alonso et al., 2015). Incorporating the annotator ranking scores, we propose *AnnoHard* to tackle the annotator's agreement with the aggregative labels and *AnnoSoft* using the cross entropy values and the probabilities of the aggregative labels over the annotators.

In this shared task, we are given four datasets for individual classification tasks in various scenarios: different languages (i.e., Arabic and English) and text formats (i.e., short texts and dialogues). We used only the provided datasets during the practice and evaluation phases and followed the official splits. No additional training data was introduced to boost the performance. The detail of the datasets can be found in Section 4.

3 Methods

Subjective tasks rely heavily on the annotators' personal opinions and their ability to perform a specific task. We focus on a particular case of a mixed set of annotators without any pre-knowledge about the annotator (e.g., background and expertise). In this case, not all instances share the same set of annotators. For example, instance #1 is annotated by annotators 1, 2, and 3; instance #2 is annotated by annotators 2, 3, and 4; and instance #3 is annotated by annotators 3, 6, 7, and 8 etc. For simplicity, we represent each instance using the term frequency-inverse document frequency (TF-IDF) or pre-trained word embeddings. The detail of the features we selected for the experiments can be found in Section 4.3. To incorporate the performance of the different sets of annotators, the first step is to compute each annotator's ranking score on a specific task (Section 3.1). Second, we combine the ranking scores of all annotators working on the same training instance. Then, we adjust the training instance weights by the sum of the annotator ranking scores (Section 3.2).

3.1 Annotator Ranking Scores

Considering a classification problem, we define the problem as $f(x)$ with an input x to predict a label y. Given a dataset with n instances $\mathcal{D} = \{d_1, d_2, ..., d_n\}$, assume that we have m annotators hired for working on this task $\mathcal{A} = \{A_1, A_2, ..., A_m\}$. For each instance d, a set of h annotators A_d is assigned to label the instance where $A_d \subseteq \mathcal{A}$, $h \in [2, m]$ and $\neg\forall A_d \neq A'_d$. During the annotation process, a set of labels $\mathcal{L}_{\text{hard}} = \{l_1, l_2, ..., l_n\}$ is provided by aggregating the labels of the majority, and these labels are referred to as *hard labels*. The probabilities of each class being agreed by the h annotators are referred to as *soft labels*. For example, if we have a classification task to predict a class either 0 or 1, 3 out of 5 annotators voted 0 and 2 out of 5 annotators voted 1, then the hard label is 1, and the soft label is $[0.6, 0.4]$. To incorporate the agreements

[1] https://github.com/summer1278/SemEval23-11-Diagreements

among the annotators, two functions are proposed to compute the rank of the annotators on a task based on the hard and soft labels. The majority of agreements among the annotators on the instances drive hard labels. Assuming we trust the majority, a straightforward way to compute the annotator ranking score is to compare their agreement with the majority. We refer to the ratio of matching majority agreements as *AnnoHard* α,

$$\alpha = \frac{N_{match}}{N_p}, \tag{1}$$

where N_{match} is the number of the annotator's label matches to the *hard label*, and N_p is the number of instances the annotator participates in.

In contrast, soft labels P are driven by the probabilities of obtaining a certain class for this instance. Using binary classification as an example for its simplicity, *AnnoSoft* β is defined by,

$$\beta = \frac{\sum_{N_{match}} \max(P_+, P_-)}{N_{match}}, \tag{2}$$

where P_+ is the probability of getting a positive label among the annotators, and P_- is the probability of getting a negative label among the annotators when an annotator's label matches the *hard label*. As mentioned, Eq. 3.1 is based on the assumption of binary classification, and it can be further extended to $P_1, P_2 ... P_z$ for z-class classification. The idea behind this function is to map the level of the majority who agreed to the reliability of the annotator. Therefore, this function gives a higher score when the annotator agrees with a label with more other annotators.

3.2 Instance Weighting

In supervised learning, instance weights ensure that each observation is given a weight to reflect its *importance* to the training. Following the studies in cost-sensitive classification (Plank et al., 2014; Alonso et al., 2015), we update the weight of the instance C using the sum of the ranking scores of the participating annotators for each instance. For example, to solve the primary problem of the Support Vector Machine (SVM) (Solon et al., 2015), the cost-sensitive loss function is defined by,

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i, \tag{3}$$

where C is a cost variable. In this particular case, C is computed by the sum of *AnnoHard* $\sum_{j=1}^{h} \alpha_j$ or *AnnoSoft* $\sum_{j=1}^{h} \beta_j$ for each instance. It can also be applied to other cost-sensitive classifiers such as Random Forest and Multi-layer Perceptron (MLP).

Table 1: Data splits in Le-Wi-Di dataset. #train and #dev denotes the number of training and development instances, respectively.

dataset	#train	#dev
ArMIS	657	141
ConvAbuse	2398	812
HS-Brexit	784	168
MD-Agreement	6592	1104

4 Experiments

The system was developed using the Le-Wi-Di dataset (Leonardellli et al., 2023), which includes 4 sub datasets: ArMIS (Almanea and Poesio, 2022), ConvAbuse (Cercas Curry et al., 2021), HS-Brexit (Akhtar et al., 2021) and MD-Agreement (Leonardelli et al., 2021). The statistics of the train or development split for each dataset can be found in Table 1. Individual models were trained on each dataset. The same feature selection and training strategies were applied to all datasets. In this section, we state the details of the evaluation metrics (Section 4.1), preprocessing (Section 4.2), document representation and training strategy (Section 4.3), and performance evaluation (Section 4.4). We conducted a series of experiments during the practice phase for the training step.

4.1 Evaluation Metrics

Following the description of the task, we use two measures to evaluate the performance of the developed system. Micro F1 score F_1 is used to evaluate the performance on hard labels using the number of True Positives (TP), False Positives (FP) and False Negatives (FN):

$$F_1 = \frac{\mathrm{TP}}{\mathrm{TP} + \frac{1}{2} \cdot (\mathrm{FP} + \mathrm{FN})} \tag{4}$$

A model with a higher F_1 indicates better performance on hard evaluation. Rather than the hard evaluation, we evaluate the model's performance on soft labels using cross entropy H computed by the predicted probabilities of each label y^* and the expected target label y,

$$H = -\sum_{i=1}^{N} \frac{1}{N} y \log(y^*), \tag{5}$$

where N is the number of test instances. A model that performs well on soft evaluation obtains a lower H.

4.2 Preprocessing

Three datasets (ArMIS, HS-Brexit and MD-Agreement) were collected from social media platforms. Hence, we removed HTML markups, URLs, hashtags, @names, punctuation, non-ASCII digits and extra white spaces. For all datasets, we used NLTK Toolkit[2] to stem, tokenize the instances into words and convert them into bi-grams. To develop a method with generality in the task settings, we did not consider any extra information about the annotators or the datasets.

4.3 Training

Pre-trained word embeddings have shown promising performance on several classification tasks. We use the Smoothed Inversed Frequency (SIF) (Arora et al., 2017) to present each instance by the weighted average of the word embeddings. We used TF-IDF (tfidf) computed by 3,000 most frequently occurring bi-grams as a baseline and compared the results with a collection of pre-trained word embeddings: FastText trained on Common Crawl[3] and Wikipedia News Corpus[4] (crawl and news) (Mikolov et al., 2018), Extended Dependency Skipgram (extvec) (Komninos and Manandhar, 2016), GloVe (glove) (Pennington et al., 2014), Skip-gram (twitter) (Mikolov et al., 2013) and Turian (turian) (Turian et al., 2010). For the ArMIS dataset, due to the availability of Arabic word embedding models, we only compared the FastText (ar) with TF-IDF. We use the implementation from flair (Akbik et al., 2019).

We conduct a comprehensive study on 56 possible feature and algorithm combinations on each sub-task to find the best combination. We randomly select 70% for training and 30% for validation from the given train split. We validate the classification model using various learning algorithms: Bernoulli Naïve Bayes (BernoulliNB), Gaussian Naïve Bayes (GaussianNB), Multi-layer Perceptron (MLP), Logistic Regression (LR), Random Forest (RF), Extra Trees, Linear Support Vector Machine with Stochastic Gradient Descent (SVM) and K-Nearest Neighbors (KNN). The hyperparameters are tuned by gird search. To reduce the negative effects of imbalanced datasets, we apply a simple oversampling technique that replicates the instances from the minority class using scikit-learn[5].

Using MD-agreement as an example, Table 3 shows the top 10 combinations of features and learning algorithms by F_1. The model using TF-IDF with Random Forest shows the best performance. As a popular traditional learning algorithm, Random Forest benefits the development from its simplicity and relatively short time to train or test. Similar results were also found in all short text datasets and the methods using annotator rank functions. One exception is the ConvAbuse dataset (see Appendix Table 9), which contains conversation dialogues between two people. Table 9 shows that the top 8 models use MLP or Random Forest. FastText with MLP offers slightly better performance than

Table 2: Top 10 models (feature and learning algorithm) without instance weighting (baseline) on the ArMIS dataset by F_1 in descending order. b_acc denotes the class-balanced accuracy and F_1 denotes the Micro F1 score. train_dur and test_dur denote the train time and test time in seconds, respectively.

model (feat-alg)	b_acc	F_1	train_dur	test_dur
tfidf-RandomForest	0.6652	0.7121	0.1829	0.0115
tfidf-BernoulliNB	0.6833	0.7020	0.0050	0.0024
tfidf-MLP	0.6686	0.6869	7.8901	0.0025
tfidf-LR	0.6605	0.6818	0.4438	0.0008
tfidf-ExtraTrees	0.6443	0.6717	0.1202	0.0117
tfidf-SVM	0.6462	0.6515	0.0399	0.0018
ar-MLP	0.6306	0.6465	2.3133	0.0006
ar-KNN	0.6211	0.6465	0.0004	0.0392
ar-RandomForest	0.5745	0.6364	0.1835	0.0105
ar-ExtraTrees	0.5726	0.6364	0.0804	0.0108

Table 3: Top 10 models without instance weighting (baseline) on the MD-Agreement dataset by F_1 in descending order.

model (feat-alg)	b_acc	F_1	train_dur	test_dur
tfidf-RandomForest	0.6466	0.7695	1.5393	0.0341
tfidf-ExtraTrees	0.6640	0.7685	4.3721	0.0365
crawl-MLP	0.7062	0.7679	34.1913	0.0711
news-MLP	0.6690	0.7679	115.8468	0.0877
twitter-MLP	0.6395	0.7457	25.0194	0.0290
glove-MLP	0.6565	0.7422	21.0911	0.0278
tfidf-LR	0.7069	0.7401	2.0270	0.0110
crawl-RandomForest	0.5702	0.7376	0.6590	0.0197
tfidf-MLP	0.6823	0.7371	75.3164	0.0283
news-RandomForest	0.5647	0.7341	0.6646	0.0197

[2] https://www.nltk.org/
[3] https://commoncrawl.org/
[4] https://autonlp.ai/datasets/wikipedia-news-corpus
[5] https://scikit-learn.org/stable/

TF-IDF when dealing with conversations. ArMIS is the only dataset in this shared task that consists of text from a low-resource language, Arabic. We found the top 6 models using TF-IDF as the feature vectors.

4.4 Results using Annotator Ranking Scores

In this section, we report the results using the official train split to develop the model and validate it on the development split. As we consider a sub-problem of learning with disagreements, we focus on evaluating the two datasets with mixed sets of annotators: ConvAbuse (i.e., conversations) and MD-Agreement (i.e., short texts). Table 4 shows the evaluation results on the hard (F_1) and soft labels (H). In the ConvAbuse dataset, using annotator rank weighting (both *AnnoHard* and *AnnoSoft*) improves model performance on hard labels compared to the baseline method. With *AnnoSoft*, the model improves the performance on soft evaluation while *AnnoHard* shows a performance drop. *AnnoSoft* is computed by soft labels, which contain the contributions from the other annotators on each instance they worked on. In contrast, the computation of *AnnoHard* only considers the individual agreements with the majority labels on a specific task. This ranking function is independent of the distribution of the other annotators. For example, when some annotators work on a small number of texts perfectly, they would obtain extremely high ranking scores using *AnnoHard*. This case is undesirable and would cause further problems of human label variation (Plank, 2022). In the MD-Agreement dataset, using *AnnoSoft* improves the performance on soft and hard evaluation as in the ConvAbuse dataset. However, we observe the performance drop using *AnnoHard*. We suspect it is caused by the same reason, and short texts are more sensitive to the introduced costs during training.

ArMIS and HS-Brexit were labelled by the same annotators for all instances throughout the dataset, which is not the main focus of the proposed method. We find the proposed method suffers from the negative effects of over-fitting the datasets (see Appendix Section B).

Table 4: Micro F1 Score F_1 and Cross Entropy H on the ConvAbuse and MD-Agreement datasets using *AnnoHard* α and *AnnoSoft* β. w/ $\cdot$ denotes a particular weighting function of the annotators' rank.

dataset	weighting	F_1	H
ConvAbuse	baseline	0.8645	2.8369
	w/ α	0.9581	2.9644
	w/ β	0.9631	2.7634
MD-Agreement	baseline	0.8324	6.5013
	w/ α	0.8179	6.6696
	w/ β	0.8397	6.4212

5 Limitations and Future Directions

Due to the time limitation, this system was developed using traditional machine learning algorithms or an MLP. The proposed solution is limited to a specific scenario of multiple annotators' disagreements: various sets of annotators for labelling each instance in a dataset. The main contributions are the proposed heuristic methods to compute the rank of the annotator driven by the soft and hard labels.

AnnoHard is sensitive to the workload of the annotators, as stated in the previous section. It sometimes fails to reflect the annotator's ability to perform a specific task with a light workload. Hence, it is limited to a sizeable amount of data labelled by an annotator. An alternative strategy might be introducing a constraint about the workload of the annotators to the ranking function. In this case, the annotators labelling more instances following the majority would be encouraged, whereas those with fewer labelled instances would be downgraded.

On the other hand, model training was carried out using hard labels, introducing a variable of the sum of the annotator ranking scores for each instance. The feature and algorithm selection was based on the hard labels as well. We could train the model and select the best candidate from these combinations using the soft labels.

Given these limitations in the proposed method, we hope to encourage the community to explore further the idea of using the interpretability linking to the heuristics.

6 Conclusions

In this paper, we proposed two heuristic functions to compute the annotator ranking scores. We used cost-sensitive learning algorithms and introduced a cost variable for each instance into the training step. The variable is computed by the sum of the ranking scores of the participating annotators for each instance. We discussed the advantages of using Random Forest as the learning algorithm for the short text datasets and the limitations of the proposed methods. Compared to a typical classifier, we observed slight improvements in the soft and hard evaluation with the proposed function *AnnoSoft*.

Acknowledgements

We thank the task organizers for the arrangements and the anonymous reviewers for carefully reading our manuscript and for their insightful comments and suggestions.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Sohail Akhtar, Valerio Basile, and Viviana Patti. 2020. Modeling annotator perspective and polarized opinions to improve hate speech detection. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, 8(1):151–154.

Sohail Akhtar, Valerio Basile, and Viviana Patti. 2021. Whose opinions matter? perspective-aware models to identify opinions of hate speech victims in abusive language detection. *arXiv preprint arXiv:2106.15896*.

Dina Almanea and Massimo Poesio. 2022. ArMIS - the Arabic misogyny and sexism corpus with annotator subjective disagreements. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2282–2291, Marseille, France. European Language Resources Association.

Héctor Martínez Alonso, Barbara Plank, Arne Skjærholt, and Anders Søgaard. 2015. Learning to parse with iaa-weighted loss. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1357–1361.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of 5th International Conference on Learning Representations (ICLR)*.

Amanda Cercas Curry, Gavin Abercrombie, and Verena Rieser. 2021. ConvAbuse: Data, analysis, and benchmarks for nuanced abuse detection in conversational AI. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7388–7403, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Peter Geibel and Fritz Wysotzki. 2003. Perceptron based learning with example dependent and noisy costs. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, page 218–225. AAAI Press.

Shohei Higashiyama, Kazuhiro Seki, and Kuniaki Uehara. 2013. Clinical entity recognition using cost-sensitive structured perceptron for ntcir-10 mednlp. In *Proceedings of the 10th NTCIR Conference*.

Alexandros Komninos and Suresh Manandhar. 2016. Dependency based embeddings for sentence classification tasks. In *NAACL-HLT*, pages 1490–1500, San Diego, California. Association for Computational Linguistics.

Elisa Leonardelli, Stefano Menini, Alessio Palmero Aprosio, Marco Guerini, and Sara Tonelli. 2021. Agreeing to disagree: Annotating offensive language datasets with annotators' disagreement. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10528–10539, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Elisa Leonardellli, Gavin Abercrombie, Dina Almanea, Valerio Basile, Tommaso Fornaciari, Barbara Plank, Massimo Poesio, Verena Rieser, and Alexandra Uma. 2023. SemEval-2023 Task 11: Learning With Disagreements (LeWiDi). In *Proceedings of the 17th International Workshop on Semantic Evaluation*, Toronto, Canada. Association for Computational Linguistics.

Dennis V Lindley. 1958. Fiducial distributions and bayes' theorem. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 102–107.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Barbara Plank. 2022. The "problem" of human label variation: On ground truth in data, modeling and evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi. Association for Computational Linguistics.

Barbara Plank, Dirk Hovy, and Anders Søgaard. 2014. Learning part-of-speech taggers with inter-annotator agreement loss. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 742–751, Gothenburg, Sweden. Association for Computational Linguistics.

Fabio Poletto, Valerio Basile, Manuela Sanguinetti, Cristina Bosco, and Viviana Patti. 2021. Resources and benchmark corpora for hate speech detection: a systematic review. *Language Resources and Evaluation*, 55:477–523.

Hendrik Schuff, Jeremy Barnes, Julian Mohme, Sebastian Padó, and Roman Klinger. 2017. Annotation, modelling and analysis of fine-grained emotions on a stance and sentiment detection corpus. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 13–23, Copenhagen, Denmark. Association for Computational Linguistics.

Gary Solon, Steven J Haider, and Jeffrey M Wooldridge. 2015. What are we weighting for? *Journal of Human resources*, 50(2):301–316.

Anja Thieme, Danielle Belgrave, and Gavin Doherty. 2020. Machine learning in mental health: A systematic review of the hci literature to support the development of effective and implementable ml systems. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 27(5):1–53.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394.

A Discussion on Feature Selection and Learning Algorithms

We present the top 10 models on the MD-agreement dataset with *AnnoHard* and *AnnoSoft* in Tables 5 and 6, respectively. We find that both methods show the best performance on hard labels using TF-IDF with Random Forest, which followed the results without using instance weighting. Extra Trees performs closely to Random Forest, due to its complexity, it spends more than twice of train time using Random Forest.

Furthermore, we present the results on the MD-agreement (Table 8) and ConvAbuse (Table 9) datasets using all 56 possible combinations of features and algorithms. We observe Random Forest, Extra Trees and MLP perform consistently well in all given datasets. Tree-based classifiers (i.e., Random Forest and Extra Trees) use a series of conditional statements to partition the train set into subsets. Then, the successive branches contribute to the model training. These classifiers are particularly good at handling complex relationships among features. Whereas a Naïve Bayes (NB) classifier (either GaussianNB or BernoulliNB) uses the Bayes' Theorem (Lindley, 1958) and assumes all extracted features are independent. The words in subjective tasks are typically dependent on surrounding words and sentences (i.e., context). It explains the worse performance using an NB classifier for these tasks.

Table 5: Top 10 models with *AnnoHard* on the MD-Agreement dataset by F_1 in descending order.

model (feat-alg)	b_acc	F_1	train_dur	test_dur
tfidf-RandomForest	0.6469	0.7685	1.7174	0.0340
tfidf-ExtraTrees	0.6645	0.7685	4.5001	0.0486
crawl-RandomForest	0.5783	0.7422	0.5965	0.0174
tfidf-LR	0.7080	0.7396	1.4833	0.0046
news-RandomForest	0.5639	0.7336	0.6018	0.0168
crawl-LR	0.7098	0.7290	0.4116	0.0013
twitter-RandomForest	0.5575	0.7280	0.4226	0.0160
crawl-ExtraTrees	0.5459	0.7275	0.1774	0.0186
glove-RandomForest	0.5556	0.7240	0.4052	0.0162
extvec-RandomForest	0.5508	0.7235	0.6101	0.0163

B Evaluation on the ArMIS and HS-Brexit Datasets

In this section, we discuss the results of hard and soft evaluations on the ArMIS and HS-Brexit datasets. These two datasets have the same set of annotators (i.e., 3 and 6 annotators) for all instances within them. Similar to the other two datasets, we applied the weights by computing the two proposed annotator rank functions. Given that the weights were not normalised, we increased the scale of weights for the instances. For example, in the ArMIS dataset, the weights of instances in the baseline method are [1, 1, 1, ..., 1]. However, they have increased to [2.66, 2.66,...,2.66] and [2.75, 2.75,...,2.75] with α and β, respectively. In Table 7, we observe a performance drop in hard and soft scores using both functions. The developed models apparently suffer from the over-fitting problem. They are not suitable to be used in this scenario.

Table 6: Top 10 models with *AnnoSoft* on the MD-Agreement dataset by F_1 in descending order.

model (feat-alg)	b_acc	F_1	train_dur	test_dur
tfidf-RandomForest	0.6517	0.7725	1.6401	0.0362
tfidf-ExtraTrees	0.6618	0.7674	4.5740	0.0397
tfidf-LR	0.7093	0.7422	1.1357	0.0049
crawl-RandomForest	0.5722	0.7391	0.6181	0.0172
news-RandomForest	0.5664	0.7351	0.5920	0.0165
twitter-RandomForest	0.5628	0.7300	0.4342	0.0174
crawl-LR	0.7094	0.7285	0.6852	0.0030
glove-RandomForest	0.5572	0.7235	0.4114	0.0164
extvec-RandomForest	0.5493	0.7235	0.6398	0.0182
news-ExtraTrees	0.5336	0.7199	0.1825	0.0169

Table 7: Micro F1 Score F_1 and Cross Entropy H on the ArMIS and HS-Brexit datasets using *AnnoHard* α and *AnnoSoft* β.

dataset	weighting	F_1	H
ArMIS	baseline	0.7376	8.3240
	w/ α	0.6454	7.5047
	w/ β	0.7234	8.3239
HS-Brexit	baseline	0.9464	1.9735
	w/ α	0.9702	0.6167
	w/ β	0.9702	0.6167

Table 8: 56 tested combinations on the MD-Agreement dataset by F_1 in descending order.

model (feat-alg)	b_acc	F_1	train_dur	test_dur
tfidf-RandomForest	0.6466	0.7695	1.5393	0.0341
tfidf-ExtraTrees	0.6640	0.7685	4.3721	0.0365
crawl-MLP	0.7062	0.7679	34.1913	0.0711
news-MLP	0.6690	0.7679	115.8468	0.0877
twitter-MLP	0.6395	0.7457	25.0194	0.0290
glove-MLP	0.6565	0.7422	21.0911	0.0278
tfidf-LR	0.7069	0.7401	2.0270	0.0110
crawl-RandomForest	0.5702	0.7376	0.6590	0.0197
tfidf-MLP	0.6823	0.7371	75.3164	0.0283
news-RandomForest	0.5647	0.7341	0.6646	0.0197
extvec-MLP	0.6898	0.7326	38.9783	0.0822
crawl-LR	0.7115	0.7300	0.6917	0.0014
crawl-ExtraTrees	0.5502	0.7295	0.2084	0.0197
crawl-SVM	0.6455	0.7260	0.2245	0.0025
twitter-RandomForest	0.5567	0.7255	0.4611	0.0189
twitter-ExtraTrees	0.5392	0.7230	0.1689	0.0191
glove-RandomForest	0.5551	0.7219	0.4494	0.0196
extvec-RandomForest	0.5469	0.7214	0.6960	0.0192
news-ExtraTrees	0.5357	0.7214	0.2081	0.0194
news-LR	0.7082	0.7179	0.5862	0.0016
glove-ExtraTrees	0.5371	0.7179	0.1725	0.0189
tfidf-SVM	0.6129	0.7159	1.2825	0.0205
extvec-ExtraTrees	0.5239	0.7139	0.2067	0.0203
tfidf-BernoulliNB	0.6498	0.7128	0.0556	0.0295
tfidf-KNN	0.5232	0.7053	0.0072	0.4995
turian-RandomForest	0.5149	0.7032	0.3720	0.0197
turian-ExtraTrees	0.5052	0.7027	0.1559	0.0188
twitter-LR	0.6853	0.7022	0.7568	0.0006
extvec-SVM	0.6312	0.7017	0.3005	0.0026
crawl-BernoulliNB	0.6784	0.6946	0.0156	0.0085
extvec-LR	0.6879	0.6941	0.7202	0.0012
turian-MLP	0.5475	0.6886	40.8466	0.0193
turian-BernoulliNB	0.5410	0.6815	0.0031	0.0014
glove-SVM	0.6046	0.6795	0.0887	0.0090
news-SVM	0.5867	0.6749	0.3059	0.0028
glove-LR	0.6576	0.6653	0.7752	0.0013
news-BernoulliNB	0.6541	0.6653	0.0153	0.0085
twitter-BernoulliNB	0.6038	0.6633	0.0053	0.0027
crawl-KNN	0.5964	0.6557	0.0017	0.2758
news-KNN	0.5965	0.6517	0.0017	0.3063
glove-BernoulliNB	0.6053	0.6441	0.0090	0.0032
twitter-KNN	0.5782	0.6431	0.0010	0.2504
extvec-KNN	0.5882	0.6421	0.0012	0.2951
turian-SVM	0.5393	0.6365	0.0619	0.0007
glove-KNN	0.5635	0.6314	0.0010	0.2676
twitter-SVM	0.5284	0.6309	0.1056	0.0010
extvec-BernoulliNB	0.6254	0.6168	0.0143	0.0074
turian-KNN	0.5316	0.6148	0.0009	0.2168
twitter-GaussianNB	0.6343	0.5956	0.0025	0.0012
turian-LR	0.5844	0.5839	0.4493	0.0005
glove-GaussianNB	0.6210	0.5768	0.0026	0.0012
news-GaussianNB	0.6317	0.5748	0.0055	0.0036
crawl-GaussianNB	0.6278	0.5693	0.0056	0.0031
extvec-GaussianNB	0.6147	0.5536	0.0061	0.0039
turian-GaussianNB	0.5896	0.5293	0.0018	0.0007
tfidf-GaussianNB	0.5632	0.5061	0.0984	0.0424

Table 9: 56 tested combinations on the ConvAbuse dataset by F_1 in descending order.

model (feat-alg)	b_acc	F_1	train_dur	test_dur
crawl-MLP	0.7427	0.8806	35.2578	0.0328
tfidf-MLP	0.7246	0.8792	38.1889	0.0106
twitter-MLP	0.6748	0.8764	8.9885	0.0110
news-MLP	0.7600	0.8750	39.5826	0.0323
glove-MLP	0.6672	0.8694	1.4798	0.0009
crawl-RandomForest	0.6078	0.8681	0.2526	0.0160
extvec-MLP	0.7413	0.8667	36.4785	0.0322
news-RandomForest	0.6035	0.8667	0.2522	0.0164
tfidf-SVM	0.6570	0.8639	0.1506	0.0064
tfidf-RandomForest	0.6114	0.8625	0.2348	0.0182
tfidf-LR	0.7957	0.8597	0.8683	0.0028
glove-RandomForest	0.5744	0.8583	0.2137	0.0136
crawl-ExtraTrees	0.5702	0.8569	0.1070	0.0151
news-ExtraTrees	0.5659	0.8556	0.1098	0.0151
glove-ExtraTrees	0.5616	0.8542	0.0932	0.0127
tfidf-ExtraTrees	0.6021	0.8528	0.2886	0.0199
twitter-RandomForest	0.5573	0.8528	0.2087	0.0134
twitter-ExtraTrees	0.5505	0.8528	0.0931	0.0141
extvec-RandomForest	0.5531	0.8514	0.2539	0.0161
extvec-ExtraTrees	0.5462	0.8514	0.1098	0.0149
news-SVM	0.7159	0.8472	0.0632	0.0010
glove-SVM	0.6160	0.8472	0.0257	0.0005
twitter-SVM	0.6901	0.8444	0.0168	0.0050
extvec-KNN	0.6213	0.8444	0.0007	0.0759
crawl-LR	0.8297	0.8417	0.4708	0.0005
turian-RandomForest	0.5197	0.8417	0.2110	0.0145
crawl-SVM	0.7006	0.8389	0.0488	0.0011
turian-ExtraTrees	0.5077	0.8389	0.0953	0.0140
extvec-SVM	0.7342	0.8375	0.0815	0.0010
glove-KNN	0.5991	0.8361	0.0005	0.0638
news-KNN	0.5991	0.8361	0.0007	0.0753
turian-MLP	0.5198	0.8361	5.3387	0.0063
turian-BernoulliNB	0.4983	0.8347	0.0012	0.0003
tfidf-KNN	0.5595	0.8333	0.0028	0.1304
crawl-KNN	0.5985	0.8236	0.0007	0.0829
twitter-KNN	0.5985	0.8236	0.0005	0.0682
glove-BernoulliNB	0.5537	0.8236	0.0017	0.0008
twitter-BernoulliNB	0.6012	0.8167	0.0018	0.0008
twitter-LR	0.8166	0.8139	0.5157	0.0005
turian-SVM	0.5669	0.8111	0.0091	0.0003
extvec-LR	0.8022	0.8014	0.4472	0.0005
extvec-BernoulliNB	0.6980	0.8000	0.0042	0.0023
turian-KNN	0.5698	0.7986	0.0004	0.0707
glove-LR	0.8153	0.7944	0.2059	0.0003
news-LR	0.7888	0.7847	0.4394	0.0006
tfidf-GaussianNB	0.6118	0.7306	0.0350	0.0128
crawl-BernoulliNB	0.7428	0.7250	0.0049	0.0027
news-BernoulliNB	0.7242	0.7111	0.0048	0.0026
turian-GaussianNB	0.6270	0.6694	0.0010	0.0003
news-GaussianNB	0.7301	0.6403	0.0022	0.0011
turian-LR	0.6458	0.6375	0.3148	0.0002
crawl-GaussianNB	0.7329	0.6333	0.0022	0.0011
tfidf-BernoulliNB	0.7019	0.5931	0.0210	0.0096
glove-GaussianNB	0.6588	0.5611	0.0012	0.0005
extvec-GaussianNB	0.6693	0.5500	0.0021	0.0011
twitter-GaussianNB	0.6871	0.5278	0.0012	0.0005