Can we get smarter than majority vote? Efficient use of individual rater's labels for content moderation

Changho Shin[*]
Department of Computer Science
University of Wisconsin Madison
1210 W Dayton St, Madison, WI 53706
cshin23@wisc.edu

Alice Schoenauer Sebag
Twitter
1355 Market St, San Francisco, CA 94103
asebag@twitter.com

Abstract

A large number of natural language processing (NLP) datasets contain crowdsourced labels. Training set labels are usually generated using majority vote from individual rater's labels, which discards a significant amount of information. This paper focuses on improving data-efficiency when training a model for "marginally abusive" Tweet classification. We compare majority vote to two families of alternative methods, changing the training process in two different steps: (1) aggregating individual labels using weak supervision to improve the quality of labels for model training, and (2) predicting individual labels using the multi-rater models proposed by Davani et al. [2022]. We find that majority vote is a strong baseline. Dawid-Skene [Dawid and Skene, 1979] and multi-rater models are not significantly better than the baseline, and the latter tend to be more susceptible to overfit. Finally, we also identify a number of practical considerations for the practitioner, such as setting a minimum number of labels per rater, or preferring soft to hard labels.

1 Introduction

Online content moderation is an important application of NLP models, whose goal is to identify e.g. potentially harmful, misleading or spammy content. Depending on the severity level of the content, the predictions can be used to decide if it should be removed, or its visibility diminished. On platforms such as Twitter, the number of necessary decisions precludes to rely exclusively on human agents. Thus, machine learning (ML) models for identifying misleading information or toxic content are a growing area of research e.g. [Zellers et al., 2019, Juuti et al., 2020, Pavlopoulos et al., 2020]. In particular, the identification of marginally abusive behavior, that is, content which falls in the margin, is a critical problem. Identifying such content makes it possible, for example, to avoid recommending it to users who do not follow its authors[2]. It is a difficult problem, due to the heterogeneity of marginal abuse, and (thankfully) the low frequency of positive labels.

Seminal work [Snow et al., 2008] showed that NLP models can perform as well when trained with crowdsourced data as with expert-labeled data. Since then, crowdsourced data has been used to solve supervised NLP tasks ranging from part-of-speech (POS) tagging [Hovy et al., 2014] to abusive content classification [Davani et al., 2022]. Crowdsourced label variability comes both from sampling errors, which all tasks are subject to, and task subjectivity, potentially stemming from the ambiguity of the task itself, or differences in background between the raters [Dumitrache, 2015]. To account for this, a frequent solution is to use majority vote of crowdsourced labels [Wong and Paritosh, 2022]. This solution is simple, but it discards a significant share of information, such as the individual labels and their rater ids. This has spurred a large area of research: is it possible to train NLP models more

[*]This research has been conducted during his internship at Twitter

[2]See [Yee et al., 2022], section 3.1 for more details on model applications.

2nd Workshop on Efficient Natural Language and Speech Processing (NeurIPS 2022).

data efficiently, i.e., improve downstream model performance by extracting more information from a given crowdsourced dataset? Background work will be covered in section 2.

This paper focuses on answering the data-efficiency question for the classification of marginally abusive Tweets: can we leverage individual labels and improve model performance compared to training on majority votes? In a pragmatic approach to this problem, we compare two families of methods, either (1) learning aggregation rules from the data - leveraging weak supervision methods, or (2) modeling individual raters prior to aggregating them, leveraging multi-rater models proposed by Davani et al. [2022]. Our contributions are the following: (1) we obtained the first comparative results from weak supervision methods and multi-rater models on marginal abuse classification, demonstrating majority vote to be a strong baseline; and (2) we make recommendations about practical considerations for practitioners working on similar problems regarding e.g. rater filtering or the usage of soft labels.

The rest of the paper is organized as follows: related works are presented in section 2. Our dataset and methods are explained in section 3. Finally, results are presented in section 4, prior to discussing them in section 5. Further experimental results are described in Appendix C and D.

2 Related work

Crowdsourced data for NLP problems As mentioned in the introduction, it is now standard to train NLP models from crowdsourced data, prior to aggregating labels in some way [Difallah and Checco, 2021]. A first set of methods uses probabilistic graphical models to estimate an (unobserved) true label, based on the observed ones (see for example [Paun et al., 2018]). This is in particular the case for weak supervision methods, of which we will test three standard approaches in this work: Dawid-Skene [Dawid and Skene, 1979], FlyingSquid [Fu et al., 2020] and Snorkel [Ratner et al., 2017]. We refer the reader to Appendix A for more details on the three methods, and Zhang et al. [2021] for more context on weak supervision. Weak supervision methods provide theoretical consistency and convergence guarantees, proving that accuracy estimators are asymptotically consistent [Chen et al., 2019, 2021]. More recently, variants of Dawid-Skene have been proposed: Whitehill et al. [2009] integrate instance difficulty and rater-wise priors into their probabilistic model. Karger et al. [2011] uses an iterative algorithm to estimate the difficulty of the task and the accuracy of each rater. Sinha et al. [2018] propose a faster version of Dawid-Skene - up to 8x faster - using a hard version of the Expectation-Maximization (EM) algorithm.

Another avenue of methods, rather than focusing on estimating label models prior to training, incorporates it in the training step by designing weighting functions. For example, Plank et al. [2014] devise an inter-annotator agreement loss which gives higher weights to data points on whose class the annotators agree. Jamison and Gurevych [2015] try weighting data points with rater agreement, as well as additionally filtering data with low rater agreements, and only find an improvement with the latter. Fornaciari et al. [2021] add an auxiliary task to golden label prediction: predicting the distribution of crowdsourced ones. As an auxiliary loss, they test the Kullback-Leibler (KL) divergence, the inverse KL divergence, and the cross-entropy loss.

Finally, a last possible option, is to predict individual labels, prior to aggregating them. This is the idea behind the multi-rater models proposed by Davani et al. [2022], who leverage individual annotators' labels by applying multi-label/task learning.

Machine learning for content moderation There has recently been quite a lot of interest for ML for content moderation. This section will only give a few pointers, as a full review would be out of scope for this paper. Recent work focuses around two main directions: (i) improving the data efficiency of such models, and (ii) identifying additional features to text to improve performance. On the side of using limited labeled data as efficiently as possible, Davani et al. [2022] apply multi-rater models to hate speech classification. Another example is work from Juuti et al. [2020], who improve model performance using data augmentation techniques to mitigate toxic data scarcity. Training multilingual content moderation model is another way to improve their data efficiency, with lower resource languages benefiting from larger ones [Aluru et al., 2020], while potentially leveraging a multi-lingual lexicon in addition [Pamungkas and Patti, 2019]. In terms of additional features, previous work investigate if additional user-level information can improve performance [Wich et al.,

2021] - it usually does but can raise potential ethical concerns, or whether adding the context of a comment improves toxicity classification - Pavlopoulos et al. [2020] find that it doesn't.

Another important subject, which goes hand in hand with this application of ML, is that of fairness and ethics: making sure that those models are not biased towards penalizing marginalized groups [Yee et al., 2022]. Indeed, a large body of work has been published in this direction, and in particular, about the interplay between crowdsourcing, subjectivity and fairness when designing labeling tasks for content moderation, e.g. [Aroyo et al., 2019], [Arhin et al., 2021].

3 Data and methods

3.1 Dataset

The overall training dataset contains approx. 380,000 Tweets, with circa 1.9 million individual labels, sampled between January and May 2022. It contains 9.3% positive examples. The median number of raters per data point is 5. In the following and when not specified otherwise, labels from raters with less than 2,000 labels are removed from the training set. The evaluation dataset is a random sample of approx. 190,000 Tweets, split between validation and test sets chronologically, and covering the same period. The validation set contains $\sim$103,000 individual labels with 2.9% positive examples, while the test set contains $\sim$86,000 individual labels with 2.5% positive examples. When evaluating performance, the label for each Tweet is generated using majority voting.

3.2 Methods

Baseline The baseline model is a fine-tuned BERT [Devlin et al., 2018] trained on data with labels generated using majority voting. In case of a tie, the label is positive.

Weak supervision We choose three off-the-shelf weak supervision methods which are representative of their family of methods: Snorkel [Ratner et al., 2017], FlyingSquid [Fu et al., 2020], and Dawid-Skene [Dawid and Skene, 1979]. These three algorithms have distinct assumptions and solutions. Snorkel and FlyingSquid's label model assumes the conditional independence of raters given the true label, while Dawid-Skene assumes their independence. A core step for each method is estimating an accuracy parameter for each rater. Snorkel estimates accuracy parameters by forcing conditional independence on the graph structure using gradient descent, while FlyingSquid estimates accuracy parameters using the method of moments (triplet method). Dawid-Skene relies on the expectation-maximization (EM) algorithm for accuracy estimations. See Appendix A for a formal description of algorithms.

In terms of implementation, we use the Wrench package from Zhang et al. [2021]. Once aggregated labels have been computed for each Tweet, training is performed identically to the baseline.

Multi-rater models Proposed by Davani et al. [2022], multi-rater models are trained to model individual raters, as opposed to some aggregated/re-weighted labels. The authors propose two similar models in this paper, named the multi-label and the multi-task models. The difference between the two lies in the following: given r raters, for each datapoint the multi-label model predicts one output in $\{0, 1\}^r$, whereas the multi-task model learns to predict r outputs in $\{0, 1\}$. During training, only a limited number of labels are available per data point. The outputs of absent raters are masked. During inference, the average of all individual outputs is computed for the final output.

4 Results

4.1 Experimental setup

Experiments compare our baseline, to training on weak supervision labels generated by either of the three above-mentioned methods, to training one of the two multi-rater models and aggregating predictions using the majority vote. The prior of positive examples is set to be 0.1 for Snorkel and FlyingSquid, based on the percentage of positive examples in the training set. Experiments are conducted in TensorFlow [Abadi et al., 2015].

In our first set of experiments, we leverage Bayesian hyperparameter optimization as implemented by Weights & Biases [Biewald, 2020], to optimize the learning rate, the number of epochs, the mini-batch size and whether to use hard labels and binary cross-entropy, or soft labels and the Kullback-Leibler (KL) divergence. In those experiments, the number of raters is constant equal to 200. In all the other experiments, for the sake of speed, the learning rate, mini-batch size and number of epoch are constant, to observe (2) the effect of varying the number of raters in the training set, and (3) study variants of weak supervision and multi-rater models.

Table 1: Comparing the performance of majority voting (baseline) with three weak supervision methods, and two multi-raters models. The best run is selected using the validation set.

Method	ROC AUC	PR AUC
Baseline	0.9829	0.6854
Snorkel [Ratner et al., 2017]	0.9803	0.6750
FlyingSquid [Fu et al., 2020]	0.9791	0.6597
Dawid-Skene [Dawid and Skene, 1979]	0.9792	**0.6861**
Multi-label [Davani et al., 2022]	**0.9845**	0.6818
Multi-task [Davani et al., 2022]	0.9836	0.6756

4.2 Performance of selected methods with hyper-parameter optimization

The result is summarized in Table 1. We can observe that the best method in terms of precision-recall area under curve (PR AUC) is Dawid-Skene, with 0.6861 PR AUC on the test set. It is closely followed by the baseline, showing that majority vote performs well in real world settings - performing better than the two multi-rater models and the other two weak supervision methods. The best method in terms of ROC AUC is the multi-label approach from Davani et al. [2022]. Training with three different random seeds, we find that the baseline, Dawid-Skene and the multi-rater approaches do not have significantly different PR AUC results. See Appendix B for a full discussion on the significance of those results.

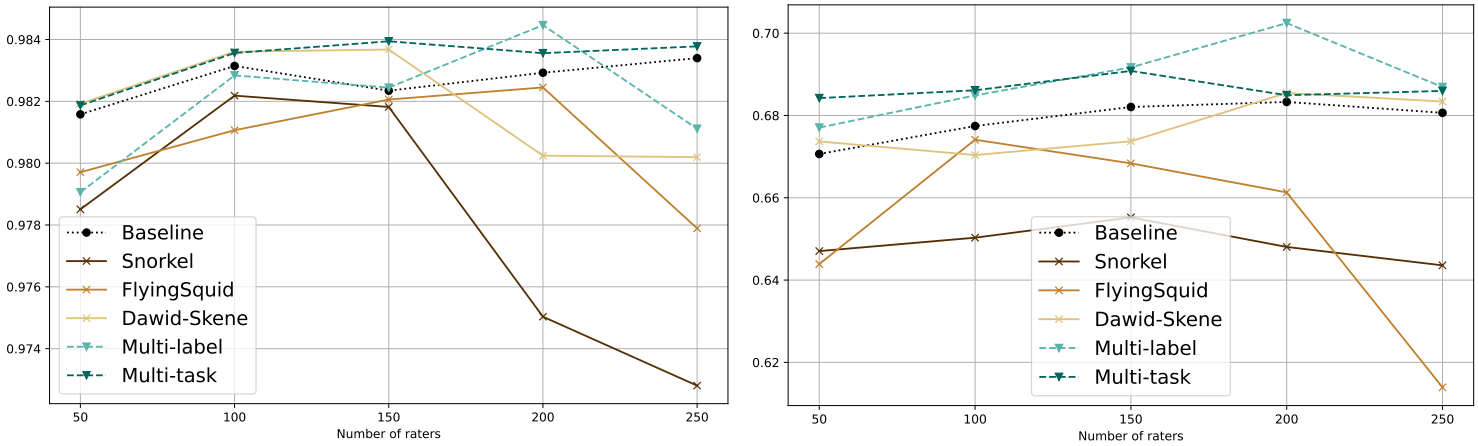
4.3 Impact of filtering out raters with smallest number of labels

Exploratory data analysis showed a correlation between the estimated individual rater accuracy from any of the three weak supervision methods, and their number of labels (Appendix C). Thus, we hypothesized that filtering out raters with less labels from the training data could be beneficial for performance. To test this hypothesis, we varied the number of raters in the dataset as [50, 100, 150, 200, 248] while keeping the other hyper-parameters constant. This corresponds to using [88%, 98%, 100%, 100%, 100%] of the dataset, thus a minimal reduction in dataset size.

The result can be found in Figure 1. We can see that testing different numbers of raters makes sense in general, as the performance of all methods changes depending on how many top raters are kept. Interestingly, no method performs best when using the full dataset, suggesting a universal rule of thumb to filter out a certain amount of least frequent raters. Focusing on PR AUC which is less saturated, we can see that the baseline, Dawid-Skene as well as the multi-label approach perform best at 200 raters. Furthermore, both FlyingSquid and Snorkel systematically perform less well than the baseline over any number of raters. Finally, we can see that multi-rater models perform systematically better than the baseline. The apparent contradiction with Table 1 can be explained by the fact that the latter results are hyper-parameter optimized, whereas the former are not. This leads to observe that multi-rater models tend to overfit the validation set more than the baseline or Dawid-Skene.

4.4 Impact of soft-labels

Previous studies [Jamison and Gurevych, 2015, Fornaciari et al., 2021] found the usage of soft labels to be beneficial to performance. To test this hypothesis, we compared performance when replacing hard binary labels and the binary cross-entropy loss, with soft labels and the KL divergence loss. For majority voting, soft labels are the ratio of positive labels for each Tweet, while weak supervision methods give soft labels based on weighted voting using each rater's accuracy parameters. Table 2



(a) ROC AUC

(b) PR AUC

Figure 1: Impact of the number of raters on performance - test set results

Table 2: Comparing the performance of majority vote and the three weak supervision methods using soft and hard labels. Test set results

Method	Label type	ROC AUC	PR AUC
Baseline	Hard	0.9829	0.6811
	Soft	0.9836	0.6832
Snorkel	Hard	0.9768	0.6491
	Soft	0.9793	0.6653
FlyingSquid	Hard	0.9826	0.6645
	Soft	0.9817	0.6713
Dawid-Skene	Hard	0.9799	0.6825
	Soft	0.9779	0.6837

shows the results. The PR AUC improves for all methods when switching from hard to soft labels: the improvement goes from less than 0.5% for the baseline and Dawid-Skene, to 2.5% for Snorkel. The ROC AUC is very little impacted on the other hand.

5 Discussion and perspectives

This paper focuses on improving data efficiency when fine-tuning a large language model for the classification of marginally abusive Tweets. We tried two different approaches to take advantage of individual rater's labels: (1) weak supervision, which estimates each rater's accuracy solely based on their rating and that of others; and (2) multi-rater modeling, which predicts individual labels prior to aggregating them. We observe that contrary to our expectations, majority voting is pretty robust. The best method - training with Dawid-Skene labels - does not significantly improve over the baseline in terms of PR AUC (Table 1 and 4). Multi-rater models display a good performance as well, but they seem more susceptible to overfit.

The performance of majority voting on marginally abusive Tweet classification is consistent with the experimental results of Wrench [Zhang et al., 2021], where majority voting turned out to be a strong baseline in many tasks. Its robustness could come from the fact that the gain from leveraging estimated rater accuracies, are smaller than the error induced by the accuracy estimation. Contrary to Dawid-Skene, Snorkel and FlyingSquid perform significantly worse than the baseline. Dawid-Skene's independence assumption between raters may be more appropriate in our setting, as opposed to the conditional independence assumption in Snorkel and FlyingSquid.

Finally, we find that (1) the number of raters in the dataset should be considered a hyper-parameter, and that filtering out a certain amount of the least frequent ones benefits performance, and (2) soft labels with a KL loss should generally be preferred to hard ones and the binary cross-entropy loss.

Further experiments on the same problem with the same dataset have shown that our experimental setup is in the large data regime: baseline performance is similar when training on 80% of the training data (data not shown). It is possible that leveraging individual rater's labels improves more significantly over majority vote in the scarce data regime. We also tried multiple approaches to improve multi-rater models, which deteriorated our metrics of interest (see Appendix D). In future work, it would be interesting to investigate which aggregation/selection method for individual predictions could improve their performance. Another direction for future work would be to compare bias severity metrics when training the baseline and multi-rater models on the same dataset: does the multi-rater approach provide additional benefits in terms of fairness?

6 Acknowledgements

The authors would like to thank Matthias Eck and Milind Ganjoo for their great suggestions and feedbacks on the project and paper. We would also like to thank the anonymous reviewers for their thoughtful feedbacks and questions.

References

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

S. S. Aluru, B. Mathew, P. Saha, and A. Mukherjee. Deep learning models for multilingual hate speech detection. *CoRR*, abs/2004.06465, 2020. URL `https://arxiv.org/abs/2004.06465`.

K. Arhin, I. Baldini, D. Wei, K. N. Ramamurthy, and M. Singh. Ground-truth, whose truth? - examining the challenges with annotating toxic text datasets. *CoRR*, abs/2112.03529, 2021. URL `https://arxiv.org/abs/2112.03529`.

L. Aroyo, L. Dixon, N. Thain, O. Redfield, and R. Rosen. Crowdsourcing subjective tasks: the case study of understanding toxicity in online discussions. In *Companion proceedings of the 2019 world wide web conference*, pages 1100–1105, 2019.

L. Biewald. Experiment tracking with weights and biases, 2020. URL `https://www.wandb.com/`. Software available from wandb.com.

M. Chen, F. Sala, and C. Ré. Lecture notes on weak supervision. 2019.

M. Chen, B. Cohen-Wang, S. Mussmann, F. Sala, and C. Ré. Comparing the value of labeled and unlabeled data in method-of-moments latent variable estimation. In *International Conference on Artificial Intelligence and Statistics*, pages 3286–3294. PMLR, 2021.

A. M. Davani, M. Díaz, and V. Prabhakaran. Dealing with disagreements: Looking beyond the majority vote in subjective annotations. *Transactions of the Association for Computational Linguistics*, 10:92–110, 2022.

A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28, 1979.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

D. Difallah and A. Checco. Aggregation techniques in crowdsourcing: Multiple choice questions and beyond. In *CIKM*, pages 4842–4844. ACM, 2021.

A. Dumitrache. Crowdsourcing disagreement for collecting semantic annotation. In F. Gandon, M. Sabou, H. Sack, C. d'Amato, P. Cudré-Mauroux, and A. Zimmermann, editors, *The Semantic Web. Latest Advances and New Domains*, pages 701–710, Cham, 2015. Springer International Publishing. ISBN 978-3-319-18818-8.

T. Fornaciari, A. Uma, S. Paun, B. Plank, D. Hovy, and M. Poesio. Beyond black & white: Leveraging annotator disagreement via soft-label multi-task learning. In *2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2021.

D. Fu, M. Chen, F. Sala, S. Hooper, K. Fatahalian, and C. Ré. Fast and three-rious: Speeding up weak supervision with triplet methods. In *International Conference on Machine Learning*, pages 3280–3291. PMLR, 2020.

D. Hovy, B. Plank, and A. Søgaard. Experiments with crowdsourced re-annotation of a POS tagging data set. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 377–382, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2062. URL `https://aclanthology.org/P14-2062`.

E. Jamison and I. Gurevych. Noise or additional information? leveraging crowdsource annotation item agreement for natural language tasks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 291–297, 2015.

M. Juuti, T. Gröndahl, A. Flanagan, and N. Asokan. A little goes a long way: Improving toxic language classification despite data scarcity. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2991–3009, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.269. URL `https://aclanthology.org/2020.findings-emnlp.269`.

D. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. *Advances in neural information processing systems*, 24, 2011.

E. W. Pamungkas and V. Patti. Cross-domain and cross-lingual abusive language detection: A hybrid approach with deep learning and a multilingual lexicon. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 363–370, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-2051. URL `https://aclanthology.org/P19-2051`.

S. Paun, B. Carpenter, J. Chamberlain, D. Hovy, U. Kruschwitz, and M. Poesio. Comparing Bayesian models of annotation. *Transactions of the Association for Computational Linguistics*, 6:571–585, 2018. doi: 10.1162/tacl_a_00040. URL `https://aclanthology.org/Q18-1040`.

J. Pavlopoulos, J. Sorensen, L. Dixon, N. Thain, and I. Androutsopoulos. Toxicity detection: Does context really matter? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4296–4305, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.396. URL `https://aclanthology.org/2020.acl-main.396`.

B. Plank, D. Hovy, and A. Søgaard. 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, 2014.

A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access, 2017.

V. B. Sinha, S. Rao, and V. N. Balasubramanian. Fast dawid-skene: A fast vote aggregation scheme for sentiment classification. *arXiv preprint arXiv:1803.02781*, 2018.

R. Snow, B. O'Connor, D. Jurafsky, and A. Ng. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii, Oct. 2008. Association for Computational Linguistics. URL https://aclanthology.org/D08-1027.

J. Whitehill, T.-f. Wu, J. Bergsma, J. Movellan, and P. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in neural information processing systems*, 22, 2009.

M. Wich, E. Mosca, A. Gorniak, J. Hingerl, and G. Groh. Explainable abusive language classification leveraging user and network data. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part V*, page 481–496, Berlin, Heidelberg, 2021. Springer-Verlag. ISBN 978-3-030-86516-0. doi: 10.1007/978-3-030-86517-7_30. URL https://doi.org/10.1007/978-3-030-86517-7_30.

K. Wong and P. K. Paritosh. k-rater reliability: The correct unit of reliability for aggregated human annotations. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 378–384. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.acl-short.42. URL https://doi.org/10.18653/v1/2022.acl-short.42.

K. Yee, A. Schoenauer-Sebag, O. Redfield, E. Sheng, M. Eck, and L. Belli. A keyword based approach to understanding the overpenalization of marginalized groups by english marginal abuse models on twitter. 2022. doi: 10.48550/ARXIV.2210.06351. URL https://arxiv.org/abs/2210.06351.

R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi. Defending against neural fake news. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/3e9f0fc9b2f89e043bc6233994dfcf76-Paper.pdf.

J. Zhang, Y. Yu, Y. Li, Y. Wang, Y. Yang, M. Yang, and A. Ratner. WRENCH: A comprehensive benchmark for weak supervision. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL https://openreview.net/forum?id=Q9SKS5k8io.

A Formal description of weak supervision methods

In this section, we provide a formal description of the three weak supervision methods we tested: Dawid-Skene [Dawid and Skene, 1979], FlyingSquid [Fu et al., 2020] and Snorkel [Ratner et al., 2017].

Problem setup We consider a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where x_i is an input data point and y_i its (unobservable) true label. A pool of rater labels each instance of $\mathcal{D}$. Given a data point x_i, the label of rater j is denoted by λ_i^j. With a little abuse of notation, denote $y \in \{0,1\}^n$ (or, $\{-1,1\}^n$ in FlyingSquid) as the true label vector, and $\lambda^j \in \{0,1\}^n$ (or, $\{-1,1\}^n$ in FlyingSquid) as rater j's label vector. Weak supervision methods define a label model, whose main goal is to estimate the most plausible pseudo-labels given (noisy) labels from a pool of raters, i.e. estimate $\hat{y} = arg\max_y P(y|\lambda^1, \cdots, \lambda^m)$. To estimate the true label y, each method estimates the accuracy of annotators with its own assumption and estimation method (cf. Table 3).

Snorkel The Snorkel algorithm is described in Algorithm 1, and its derivation can be found section 3.1. and 4.3. of Chen et al. [2019]. The gist of this algorithm is to force conditional independence between raters given a true unobservable label y, based on the graphical model. The optimization procedure contained in line 5 is guaranteed to converge since the objective function is convex.

Algorithm 1: Snorkel[Ratner et al., 2017], simplified

Input: Class prior $p_1 = P(Y=0), p_2 = P(Y=1)$, labels by annotators $\lambda^1, \cdots, \lambda^m$
Output: Estimated pseudo-label probability $\hat{y} = P(\lambda^1, \cdots, \lambda^m | Y=1)$

1
/* Learn label model $\mu = P(\lambda^i|Y)$ */
2 $L \leftarrow [\lambda^1\ \lambda^2\ \cdots\ \lambda^m]$
3 $P \leftarrow diag(p)$
4 $\Sigma_O \leftarrow L^T L/n$
5 $\hat{\mu} \leftarrow arg\min_{\mu \in \mathbb{R}^{m \times 2}} ||\Sigma_O - \mu P \mu^T||_2^2 + ||\mu P \mathbf{1} - diag(\Sigma_O)||_2^2$
6
/* Pseudo-label probability estimation */
7 **for** $i \leftarrow 1$ **to** n **do**
8 $\quad \hat{y}_i \leftarrow \dfrac{\exp(\sum_j^m \lambda_i^j \log \hat{\mu}_{j,2} + p_2)}{\sum_{k=1}^2 \exp(\sum_j^m \lambda_i^j \log \hat{\mu}_{j,k} + p_k)}$

FlyingSquid A simplified version of FlyingSquid [Fu et al., 2020] is described in Algorithm 2. The intuition for this algorithm can be obtained from the section 3.1. of Chen et al. [2019]. The core statistical assumption is the same as Snorkel's, that is, conditional independence of rater's labels given y. However, FlyingSquid relies on the so-called "triplet" method, which is a method of moments, to estimate parameters. This solves parameter estimation in linear time with guaranteed consistency.

Dawid-Skene The Dawid-Skene algorithm is described in Algorithm 3. It assumes full independence between annotators. The parameter estimation method of Dawid-Skene is a classical

Table 3: Comparison of three weak supervision methods

Method	Assumption	Accuracy Estimation Method
Snorkel	Conditional Independence	SGD: Forcing conditional independence
FlyingSquid	Conditional Independence	Triplets (Method of moments): Solve the system of equations given by conditional independence
Dawid-Skene	(Full) Independence	EM: Update estimate of Y and accuracy probability iteratively

Algorithm 2: FlyingSquid [Fu et al., 2020], simplified

Input: Class prior $p_1 = P(Y = -1), p_2 = P(Y = 1)$, labels by annotators $\lambda^1, \cdots, \lambda^m$
Output: Estimated pseudo-label probability $\hat{y} = P(\lambda^1, \cdots, \lambda^m | Y = 1)$

Initialize $A = \emptyset$
while $\{1, ..., m\} - A \neq \emptyset$ **do**
  Pick $a, b, c \in \{1, ..., m\} - A$, where $a \neq b$, $b \neq c$
  Estimate $\hat{\mathbb{E}}\left[\lambda^a \lambda^b\right] \leftarrow \frac{1}{n}\sum_{t=1}^{n} \lambda_t^a \lambda_t^b$, $\hat{\mathbb{E}}\left[\lambda^a \lambda^c\right] \leftarrow \frac{1}{n}\sum_{i=1}^{n} \lambda_i^a \lambda_i^c$, $\hat{\mathbb{E}}\left[\lambda^b \lambda^c\right] \leftarrow \frac{1}{n}\sum_{i=1}^{n} \lambda_i^b \lambda_i^c$
  $\hat{\mathbb{E}}\left[\lambda^a Y\right] \leftarrow \sqrt{|\hat{\mathbb{E}}\left[\lambda^a \lambda^b\right] \hat{\mathbb{E}}\left[\lambda^a \lambda^c\right] / \hat{\mathbb{E}}\left[\lambda^b \lambda^c\right]|}$
  $\hat{\mathbb{E}}\left[\lambda^b Y\right] \leftarrow \sqrt{|\hat{\mathbb{E}}\left[\lambda^a \lambda^b\right] \hat{\mathbb{E}}\left[\lambda^b \lambda^c\right] / \hat{\mathbb{E}}\left[\lambda^a \lambda^c\right]|}$
  $\hat{\mathbb{E}}\left[\lambda^c Y\right] \leftarrow \sqrt{|\hat{\mathbb{E}}\left[\lambda^a \lambda^c\right] \hat{\mathbb{E}}\left[\lambda^b \lambda^c\right] / \hat{\mathbb{E}}\left[\lambda^a \lambda^b\right]|}$
  $A \leftarrow A \cup \{a, b, c\}$

/* Get canonical parameters from mean parameters */
for $a \leftarrow 1$ **to** n **do**
  $\hat{\theta}_a \leftarrow BackwardMapping(\hat{\mathbb{E}}\left[\lambda^a Y\right])$
  for $b \leftarrow a + 1$ **to** n **do**
    $\hat{\theta}_{a,b} \leftarrow BackwardMapping(\hat{\mathbb{E}}\left[\lambda^a \lambda^b\right])$

/* Pseudo-label probability estimation */
for $i \leftarrow 1$ **to** n **do**
  $\hat{y_i} \leftarrow \frac{1}{Z}\exp\left(\sum_{j \neq k} \hat{\theta}_{j,k} \lambda_i^j \lambda_i^k + \sum_j \hat{\theta}_j \lambda_i^j + \log p_2\right)$

Expectation-Maximization, which estimates the true label y and updates the accuracy & error rate parameters iteratively. The EM algorithms is guaranteed to converge.

Algorithm 3: Dawid-Skene [Dawid and Skene, 1979], simplified

Input: Class prior $p_1 = P(Y = 0), p_2 = P(Y = 1)$, labels by annotators $\lambda^1, \cdots, \lambda^m$
Output: Estimated pseudo-label probability $\hat{y} = P(\lambda^1, \cdots, \lambda^m | Y = 1)$

for $i \leftarrow 1$ **to** n **do**
  $\hat{y}_i \leftarrow \sum_{j=1}^{m} \lambda_i^j / m$
$\hat{p} \leftarrow p$
/* Accuracy & error rate table $E \in \mathbb{R}^{m \times 2 \times 2}$ */
$E_{j,0,0} \leftarrow \sum_i^n (1 - \hat{y}_i)(1 - \lambda_i^j)$
$E_{j,0,1} \leftarrow \sum_i^n (1 - \hat{y}_i)\lambda_i^j$
$E_{j,1,0} \leftarrow \sum_i^n \hat{y}_i(1 - \lambda_i^j)$
$E_{j,1,1} \leftarrow \sum_i^n \hat{y}\lambda_i^j$
/* Noremalize E */
$E_{j,0,\cdot} = E_{j,0,\cdot}/(E_{j,0,0} + E_{j,0,1})$
$E_{j,1,\cdot} = E_{j,0,\cdot}/(E_{j,1,0} + E_{j,1,1})$
while *not converged* **do**
  /* E-step */
  for $i \leftarrow 1$ **to** n **do**
    $$\hat{y}_i \leftarrow \frac{\prod_{j=1}^{m} \hat{p}_2 E_{j,1,0}(1 - \lambda_i^j) E_{j,1,1} \lambda_i^j}{(\prod_{j=1}^{m} \hat{p}_1 E_{j,0,0}(1 - \lambda_i^j) E_{j,0,1} \lambda_i^j + \hat{p}_2 \prod_{j=1}^{m} E_{j,1,0}(1 - \lambda_i^j) E_{j,1,1} \lambda_i^j)}$$
  /* M-step */
  $E_{j,0,0} \leftarrow \sum_i^n (1 - \hat{y}_i)(1 - \lambda_i^j)$
  $E_{j,0,1} \leftarrow \sum_i^n (1 - \hat{y}_i)\lambda_i^j$
  $E_{j,1,0} \leftarrow \sum_i^n \hat{y}_i(1 - \lambda_i^j)$
  $E_{j,1,1} \leftarrow \sum_i^n \hat{y}\lambda_i^j$
  $\hat{p}_2 \leftarrow \sum_{i=1}^n \hat{y}_i / n$
  $\hat{p}_1 \leftarrow 1 - \hat{p}_2$

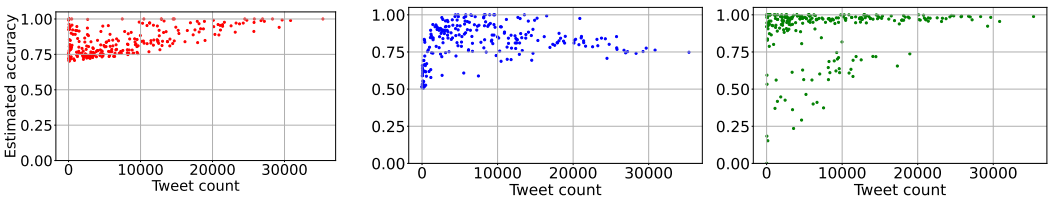
B Result significance

Table 4 compares the baseline to our methods of interest, using the average and standard deviation on three runs with different random seeds. In terms of performance, the rankings between all methods is identical to that presented in table 1. Additionally, we can see that the baseline, Dawid-Skene and the two multi-rater approaches do not have significantly different PR AUC results (additional test results not shown). This further supports the fact that majority vote is a strong baseline.

Table 4: Result significance analysis. The best run is selected using the validation set on one seed, and the results below present the average (and standard deviation) on three seeds using the same hyper-parameters. $*$: significantly different from the baseline, two-sided t-test, p-value < 0.05

Method	ROC AUC	PR AUC
Baseline	0.9824 (0.0014)	0.6889 (0.0048)
Snorkel [Ratner et al., 2017]	0.9798 (0.0008)$*$	0.6751 (0.0033)$*$
FlyingSquid [Fu et al., 2020]	0.9803 (0.0015)	0.6580 (0.0021)$*$
Dawid-Skene [Dawid and Skene, 1979]	0.9793 (0.0006)$*$	0.**6894 (0.0053)**
Multi-label [Davani et al., 2022]	0.9840 (0.0011)	0.6826 (0.0097)
Multi-task [Davani et al., 2022]	**0.9850 (0.0010)**	0.6825 (0.0036)

C Relationship between the number of Tweets a rater labeled and their accuracy estimates



(a) Snorkel (ROC AUC) (b) Snorkel (PR AUC) (c) FlyingSquid (ROC AUC)

Figure 2: Accuracy estimations of weak supervision methods

A crucial result of weak supervision methods is that we can estimate the accuracy of each rater even without access to true labels. Figure 2 shows the accuracy estimates when using all data. Here, we can observe that raters with a small number of labeled Tweet counts tend to have lower accuracy, resulting in the rater number study of subsection **??**. This phenomenon can originate from two possible reasons. The first one is that inexperienced raters may produce noisier labels. The second possible reason is that the accuracy estimation algorithm can be unstable when the number of data samples is too low for a given rater.

D Aggregation schemes for multi-rater predictions

As for individual labels, there are multiple ways to aggregate multi-rater model outputs. We tried the three following schemes:

1. **Two-step training**: after the multi-rater model has been trained, an additional dense layer is added after the last layer. This aggregation layer is trained from scratch while the multi-rater model is frozen.
2. **Double loss**: this approach also adds a layer after the last one, but trains it at the same time as the rest of network using the following loss:

$$\mathcal{L} = \mathcal{L}_r + \beta \mathcal{L}_v$$

where $\mathcal{L}$ is the total loss, $\mathcal{L}_r$ denotes the multi-rater loss and $\mathcal{L}_v$ represents the loss of the final network output against labels generated by majority voting. β is the weighting hyperparameter for two loss functions. We try $\beta = 0.1, 0.3, 1, 3$ and choose it based on the validation set performance;

3. **Weak supervision**: we apply one of the three considered weak supervision methods to aggregate the individual predictions (i.e. Snorkel, FlyingSquid, and Dawid-Skene).

The result is summarized in Table 5. Against our expectation, all three weighting methods yield worse performance than the majority vote. In particular, the performance from applying weak supervision methods is particularly low. This could be due to the fact that individual outputs are highly correlated since they share the BERT backbone. Indeed, when we observed the individual outputs of a multi-rater model, we saw that most of them were extremely close to 0 or 1.

Table 5: Weighting scheme for multi-rater models

Method	Weighting scheme	ROC AUC	PR AUC
Baseline	-	0.9829	0.6811
Multi-label	Majority voting	0.9845	0.7025
	Two step training	0.9808	0.6811
	Double loss	0.962	0.6848
	Weak supervision (Best)	0.9351	0.4597
Multi-task	Majority voting	0.9836	0.6849
	Two step training	0.9828	0.6755
	Double loss	0.9818	0.6727
	Weak Supervision (Best)	0.9350	0.4602