The Journal of Open Source Software

Learning from Crowds with Crowd-Kit

Dmitry Ustalov [*1¶], **Nikita Pavlichenko** [2], **and Boris Tseitlin** [3]

1 JetBrains, Serbia **2** Toloka, Serbia **3** Planet Farms, Portugal ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Arfon Smith

Reviewers:

- @jorgedch

Submitted: 24 September 2023
Published: unpublished

Summary

Quality control is a crux of crowdsourcing. While most means for quality control are organizational and imply worker selection, golden tasks, and post-acceptance, computational quality control techniques allow parameterizing the whole crowdsourcing process of workers, tasks, and labels, inferring and revealing relationships between them. In this paper, we present Crowd-Kit, a general-purpose crowdsourcing computational quality control toolkit. It provides efficient implementations in Python of computational quality control algorithms for crowdsourcing, including data quality estimators and truth inference methods. We focus on aggregation methods for all the major annotation tasks, from the categorical annotation in which latent label assumption is met to more complex tasks like image and sequence aggregation. We perform an extensive evaluation of our toolkit on several datasets of different natures, enabling benchmarking computational quality control methods in a uniform, systematic, and reproducible way using the same codebase. We release our code and data under an open-source license at https://github.com/Toloka/crowd-kit.

Statement of need

Means for quality control in crowdsourcing include organizational approaches, such as task design, decomposition, and golden task preparation, yet reliably automated, and computational approaches that employ relationships and statistical properties of workers, tasks, and labels. Many crowdsourcing studies of complex crowdsourcing pipelines aim to reduce their tasks to multi-classification or combine multi-classification with post-acceptance, e.g., in a seminal paper by Bernstein et al. (2010). At the same time, researchers from such fields as natural language processing, computer vision, and others develop discipline-specific methods. To be conveniently employed, these methods need to be integrated with popular data science libraries and frameworks. However, such toolkits as SQUARE (Sheshadri & Lease, 2013), CEKA (Zhang et al., 2015), Truth Inference (Zheng et al., 2017), spark-crowd (Rodrigo et al., 2019), require additional effort to be embedded in applications. We believe in addressing this issue by developing **Crowd-Kit**, an open-source production-ready Python toolkit for computational quality control in crowdsourcing. It implements popular quality control methods, providing a common ground for reliable experimentation and application. We perform an extensive evaluation of the Crowd-Kit library to provide common ground for comparisons. *In all the experiments in this paper, we used our implementations of the corresponding methods.*

Design

Our fundamental goal of Crowd-Kit development is to bridge the gap between crowdsourcing research and vivid data science ecosystem of NumPy, SciPy, pandas (McKinney, 2010), and scikit-learn (Pedregosa et al., 2011). We implemented Crowd-Kit in Python and employed the

[*]The work was done while Dmitry and Boris were with Toloka.

highly optimized data structures and algorithms available in these libraries, ensuring compatibility with the application programming interface (API) of scikit-learn and data frames/series of pandas. Even for a user not familiar with crowdsourcing but familiar with scientific computing and data analysis in Python, the basic API usage will be very straightforward:

```python
# df is a DataFrame with labeled data in form of (task, label, worker)
# gt is a Series with ground truth per task
df, gt = load_dataset('relevance-2')  # binary relevance sample dataset

# run the Dawid-Skene categorical aggregation method
agg_ds = DawidSkene(n_iter=10).fit_predict(df)  # same format as gt
```

We implemented all the methods in Crowd-Kit from scratch in Python. Although unlike spark-crowd (Rodrigo et al., 2019), our library does not provide means for running on a distributed computational cluster, it leverages efficient implementations of numerical algorithms in underlying libraries widely used in the research community. In addition to categorical aggregation methods, Crowd-Kit offers non-categorical aggregation methods, dataset loaders, and annotation quality characteristics.

Maintenance and governance

Crowd-Kit is not bound to any specific crowdsourcing platform, allowing analyzing data from any crowdsourcing marketplace (as soon as one can download the labeled data from that platform). Crowd-Kit is an open-source library working under most operating systems and available under Apache license both on GitHub and Python Package Index (PyPI).[1]

We build Crowd-Kit on top of the established open-source ecosystem and best practices. We widely use the continuous integration facilities offered by GitHub Actions for two purposes. First, every patch (*commit* in git terminology) invokes unit testing and coverage, type checking, linting, and documentation and packaging dry run. Second, every release is automatically submitted to PyPI directly from GitHub Actions via the trusted publishing mechanism to avoid potential side effects on the individual developer machines. Besides commit checks, every code change (*pull request* on GitHub) goes through code review by Crowd-Kit developers. Issues are also reported on GitHub. All code of Crowd-Kit has type annotations for additional safety and clarity. By the time of submission, our library had a test coverage of 93%.

Functionality

Crowd-Kit implements a selection of popular methods for answer aggregation and learning from crowds, dataset loaders, and annotation quality characteristics.

Aggregating and learning with Crowd-Kit

Crowd-Kit features aggregation methods suitable for most kinds of crowdsourced data, including categorical, pairwise, sequential, and image segmentation answers (see the summary in Table 1).

Methods for *categorical aggregation*, which are the most widespread in practice, assume that there is only one correct objective label per task and aim at recovering a latent true label from the observed noisy data. Some of these methods, such as Dawid-Skene and GLAD, also estimate latent parameters — aka skills — of the workers. Where the task design does not meet the latent label assumption, Crowd-Kit offers methods for aggregation *pairwise comparisons*, which are essential for subjective opinion gathering. Also, Crowd-Kit provides specialized methods for aggregating *sequences* (such as texts) and *image segmentation*. All these aggregation methods are implemented purely using NumPy, SciPy, pandas, and scikit-learn without any

[1] https://github.com/Toloka/crowd-kit & https://pypi.org/project/crowd-kit/

deep learning framework. Last but not least, Crowd-Kit offers methods for *deep learning from crowds* that learn an end-to-end machine learning model from raw responses submitted by the workers without the use of aggregation, which are available as ready-to-use modules for PyTorch (Paszke et al., 2019).

One can easily add a new aggregation method to Crowd-Kit. For example, without the loss of generality, to create a new categorical aggregator, one should extend the base class `BaseClassificationAggregator` and implement two methods, `fit()` and `fit_predict()`, filling the instance variable `labels_` with the aggregated labels.[2] Also, to add a new method for learning from crowds, one has to create a subclass from `torch.nn.Module` and implement the `forward()` method.[3]

Table 1: Summary of the implemented methods in Crowd-Kit.

Aggregation	Methods
Categorical	Majority Vote, Wawa (Appen Limited, 2021), Dawid & Skene (1979), GLAD (Whitehill et al., 2009), MACE (Hovy et al., 2013), Karger et al. (2014), M-MSR (Ma & Olshevsky, 2020)
Pairwise	Bradley & Terry (1952), noisyBT (Bugakova et al., 2019)
Sequence	ROVER (Fiscus, 1997), RASA and HRRASA (Li, 2020), Language Model (Pavlichenko et al., 2021)
Segmentation	Majority Vote, Expectation-Maximization (Jung-Lin Lee et al., 2018), RASA and HRRASA (Li, 2020)
Learning	CrowdLayer (Rodrigues & Pereira, 2018), CoNAL (Chu et al., 2021)

Dataset loaders

Crowd-Kit offers convenient dataset loaders for some popular or demonstrative datasets (see Table 2), allowing downloading them from the Internet in a ready-to-use form with a single line of code. It is possible to add new datasets in a declarative way and, if necessary, add the corresponding code to load the data as pandas data frames and series.

Table 2: Summary of the datasets provided by Crowd-Kit.

Task	Datasets
Categorical	Toloka Relevance 2 and 5, TREC Relevance (Buckley et al., 2010)
Pairwise	IMDB-WIKI-SbS (Pavlichenko & Ustalov, 2021)
Sequence	CrowdWSA (2019), CrowdSpeech (Pavlichenko et al., 2021)
Image	Common Objects in Context (Lin et al., 2014)

Annotation quality characteristics

Crowd-Kit allows one to apply commonly-used techniques to evaluate data and annotation quality, providing a unified pandas-compatible API to compute α (Krippendorff, 2018), annotation uncertainty (Malinin, 2019), agreement with aggregate (Appen Limited, 2021), Dawid-Skene posterior probability, etc.

[2] See the implementation of Majority Vote at https://github.com/Toloka/crowd-kit/blob/main/crowdkit/aggregation/classification/majority_vote.py as an example of an aggregation method.

[3] See the implementation of CrowdLayer at https://github.com/Toloka/crowd-kit/blob/main/crowdkit/learning/crowd_layer.py as an example of a method for deep learning from crowds.

Evaluation

We extensively evaluate Crowd-Kit methods for answer aggregation and learning from crowds. When possible, we compare with other authors; either way, we show how the currently implemented methods perform on well-known datasets with noisy crowdsourced data, indicating the correctness of our implementations.

Evaluation of aggregation methods

Categorical. To ensure the correctness of our implementations, we compared the observed aggregation quality with the already available implementations by Zheng et al. (2017) and Rodrigo et al. (2019). Table 3 shows evaluation results, indicating a similar level of quality as them: *D_Product*, *D_PosSent*, *S_Rel*, and *S_Adult* are real-world datasets from Zheng et al. (2017), and *binary1* and *binary2* are synthetic datasets from Rodrigo et al. (2019). Our implementation of M-MSR could not process the D_Product dataset in a reasonable time, KOS can be applied to binary datasets only, and none of our implementations handled *binary3* and *binary4* synthetic datasets, which require a distributed computing cluster.

Table 3: Comparison of the implemented categorical aggregation methods (accuracy is used).

Method	D_Product	D_PosSent	S_Rel	S_Adult	binary1	binary2
MV	0.897	0.932	0.536	0.763	0.931	0.936
Wawa	0.897	0.951	0.557	0.766	0.981	0.983
DS	0.940	0.960	0.615	0.748	0.994	0.994
GLAD	0.928	0.948	0.511	0.760	0.994	0.994
KOS	0.895	0.933	—	—	0.993	0.994
MACE	0.929	0.950	0.501	0.763	0.995	0.995
M-MSR	—	0.937	0.425	0.751	0.994	0.994

Pairwise. Table 4 shows the comparison of the *Bradley-Terry* and *noisyBT* methods implemented in Crowd-Kit to the random baseline on the graded readability dataset by Chen et al. (2013) and a larger people age dataset by Pavlichenko & Ustalov (2021).

Table 4: Comparison of implemented pairwise aggregation methods (Spearman's ρ is used).

Method	Chen et al. (2013)	IMDB-WIKI-SBS
Bradley-Terry	0.246	0.737
noisyBT	0.238	0.744
Random	−0.013	−0.001

Sequence. We used two datasets, CrowdWSA (Li & Fukumoto, 2019) and CrowdSpeech (Pavlichenko et al., 2021). As the typical application for sequence aggregation in crowdsourcing is audio transcription, we used the word error rate as the quality criterion (Fiscus, 1997) in Table 5.

Table 5: Comparison of implemented sequence aggregation methods (average word error rate is used).

Dataset	Version	ROVER	RASA	HRRASA
CrowdWSA	J1	0.612	0.659	0.676
	T1	0.514	0.483	0.500
	T2	0.524	0.498	0.520
CrowdSpeech	dev-clean	0.676	0.750	0.745

The Journal of Open Source Software

Dataset	Version	ROVER	RASA	HRRASA
	dev-other	0.132	0.142	0.142
	test-clean	0.729	0.860	0.859
	test-other	0.134	0.157	0.157

Segmentation. We annotated on Toloka a sample of 2,000 images from the MS COCO (Lin et al., 2014) dataset consisting of four object labels. For each image, nine workers submitted segmentations. The dataset is available in Crowd-Kit as `mscoco_small`. In total, we received 18,000 responses. Table 6 shows the comparison of the methods on the above-described dataset using the *intersection over union* (IoU) criterion.

Table 6: Comparison of implemented image aggregation algorithms (IoU is used).

Dataset	MV	EM	RASA
MS COCO	0.839	0.861	0.849

Evaluation of methods for learning from crowds

To demonstrate the impact of learning on raw annotator labels compared to answer aggregation in crowdsourcing, we compared the implemented methods for learning from crowds with the two classical aggregation algorithms, Majority Vote (MV) and Dawid-Skene (DS). We picked the two most common machine learning tasks for which ground truth datasets are available: text classification and image classification. For text classification, we used the IMDB Movie Reviews dataset (Maas et al., 2011), and for image classification, we chose CIFAR-10 (Krizhevsky, 2009). In each dataset, each object was annotated by three different annotators; 100 objects were used as golden tasks.

We compared how different methods for learning from crowds impact test accuracy. We picked two different backbone networks for text classification, LSTM (Hochreiter & Schmidhuber, 1997) and RoBERTa (Liu et al., 2019), and one backbone network for image classification, VGG-16 (Simonyan & Zisserman, 2015). Then, we trained each backbone in three scenarios: use the fully-connected layer after the backbone without taking into account any specifics of crowdsourcing (Base), CrowdLayer method by Rodrigues & Pereira (2018), and CoNAL method by Chu et al. (2021). Table 7 shows the evaluation results.

Table 7: Comparison of different methods for deep learning from crowds with traditional answer aggregation methods (test set accuracy is used).

Dataset	Backbone	CoNAL	CrowdLayer	Base	DS	MV
IMDb	LSTM	0.844	0.825	0.835	0.841	0.819
IMDb	RoBERTa	0.932	0.928	0.927	0.932	0.927
CIFAR-10	VGG-16	0.825	0.863	0.882	0.877	0.865

Our experiment shows that it is feasible to train a deep learning model from the raw annotated data, skipping trivial aggregation methods like MV. However, specialized methods like CoNAL and CrowdLayer or non-trivial aggregation methods such as DS can significantly enhance prediction accuracy. It is crucial to make a well-informed model selection to achieve optimal results. We believe that Crowd-Kit can seamlessly integrate these methods into machine learning pipelines that utilize crowdsourced data with reliability and ease.

The Journal of Open Source Software

Conclusion

Our experience running Crowd-Kit in production for processing crowdsourced data at Toloka shows that it successfully handles industry-scale datasets without needing a large compute cluster. We believe that the availability of computational quality control techniques in a standardized way would open new venues for reliable improvement of the crowdsourcing quality beyond the traditional well-known methods and pipelines.

Acknowledgements

We are grateful to Enrique G. Rodrigo for sharing the spark-crowd evaluation dataset. We want to thank Vladimir Losev, Evgeny Tulin, and Iulian Giliazev for their earlier contributions to the library. We received no external funding.

References

Appen Limited. (2021). *Calculating Worker Agreement with Aggregate (Wawa)*. https://success.appen.com/hc/en-us/articles/202703205-Calculating-Worker-Agreement-with-Aggregate-Wa

Bernstein, M. S., Little, G., Miller, R. C., Hartmann, B., Ackerman, M. S., Karger, D. R., Crowell, D., & Panovich, K. (2010). Soylent: A Word Processor with a Crowd Inside. *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, 313–322. https://doi.org/10.1145/1866029.1866078

Bradley, R. A., & Terry, M. E. (1952). Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, *39*(3/4), 324–345. https://doi.org/10.2307/2334029

Buckley, C., Lease, M., & Smucker, M. D. (2010). Overview of the TREC 2010 Relevance Feedback Track (Notebook). *The Nineteenth TREC Notebook*. https://www.ischool.utexas.edu/~ml/papers/trec-notebook-2010.pdf

Bugakova, N., Fedorova, V., Gusev, G., & Drutsa, A. (2019). Aggregation of pairwise comparisons with reduction of biases. *2019 ICML Workshop on Human in the Loop Learning*. https://arxiv.org/abs/1906.03711

Chen, X., Bennett, P. N., Collins-Thompson, K., & Horvitz, E. (2013). Pairwise Ranking Aggregation in a Crowdsourced Setting. *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, 193–202. https://doi.org/10.1145/2433396.2433420

Chu, Z., Ma, J., & Wang, H. (2021). Learning from Crowds by Modeling Common Confusions. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(7), 5832–5840. https://doi.org/10.1609/aaai.v35i7.16730

Dawid, A. P., & Skene, A. M. (1979). 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. https://doi.org/10.2307/2346806

Fiscus, J. G. (1997). A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER). *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, 347–354. https://doi.org/10.1109/ASRU.1997.659110

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hovy, D., Berg-Kirkpatrick, T., Vaswani, A., & Hovy, E. (2013). Learning Whom to Trust with MACE. *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1120–1130. https://aclanthology.org/N13-1132

Jung-Lin Lee, D., Das Sarma, A., & Parameswaran, A. (2018). *Quality Evaluation Methods for Crowdsourced Image Segmentation* [Technical Report]. Stanford University; Stanford InfoLab. http://ilpubs.stanford.edu:8090/1161/

Karger, D. R., Oh, S., & Shah, D. (2014). Budget-Optimal Task Allocation for Reliable Crowdsourcing Systems. *Operations Research*, *62*(1), 1–24. https://doi.org/10.1287/opre.2013.1235

Krippendorff, K. (2018). *Content Analysis: An Introduction to Its Methodology* (Fourth Edition). SAGE Publications, Inc. ISBN: 978-1-5063-9566-1

Krizhevsky, A. (2009). *Learning Multiple Layers of Features from Tiny Images*. University of Toronto. https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

Li, J. (2020). Crowdsourced Text Sequence Aggregation Based on Hybrid Reliability and Representation. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1761–1764. https://doi.org/10.1145/3397271.3401239

Li, J., & Fukumoto, F. (2019). A Dataset of Crowdsourced Word Sequences: Collections and Answer Aggregation for Ground Truth Creation. *Proceedings of the First Workshop on Aggregating and Analysing Crowdsourced Annotations for NLP*, 24–28. https://doi.org/10.18653/v1/D19-5904

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. *Computer Vision – ECCV 2014*, 740–755. https://doi.org/10.1007/978-3-319-10602-1_48

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. https://arxiv.org/abs/1907.11692

Ma, Q., & Olshevsky, A. (2020). Adversarial Crowdsourcing Through Robust Rank-One Matrix Completion. *Advances in Neural Information Processing Systems 33*, 21841–21852. https://proceedings.neurips.cc/paper/2020/file/f86890095c957e9b949d11d15f0d0cd5-Paper.pdf

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 142–150. https://aclanthology.org/P11-1015

Malinin, A. (2019). *Uncertainty Estimation in Deep Learning with application to Spoken Language Assessment* [PhD thesis, University of Cambridge]. https://doi.org/10.17863/CAM.45912

McKinney, W. (2010). Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*, 56–61. https://doi.org/10.25080/Majora-92bf1922-00a

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, *32*. https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

Pavlichenko, N., Stelmakh, I., & Ustalov, D. (2021). CrowdSpeech and Vox DIY: Benchmark Dataset for Crowdsourced Audio Transcription. *Proceedings of the*

Neural Information Processing Systems Track on Datasets and Benchmarks. https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/65ded5353c5ee48d0b7d48c591b8f430-html

Pavlichenko, N., & Ustalov, D. (2021). IMDB-WIKI-SbS: An Evaluation Dataset for Crowdsourced Pairwise Comparisons. *NeurIPS Data-Centric AI Workshop*. https://datacentricai.org/neurips21/papers/115_CameraReady_NeurIPS_2021_Data_Centric_AI_IMDB_WIKI_SbS-2.pdf

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html

Rodrigo, E. G., Aledo, J. A., & Gámez, J. A. (2019). spark-crowd: A Spark Package for Learning from Crowdsourced Big Data. *Journal of Machine Learning Research*, *20*, 1–5. https://jmlr.org/papers/v20/17-743.html

Rodrigues, F., & Pereira, F. C. (2018). Deep Learning from Crowds. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*(1), 1611–1618. https://doi.org/10.1609/aaai.v32i1.11506

Sheshadri, A., & Lease, M. (2013). SQUARE: A Benchmark for Research on Computing Crowd Consensus. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, *1*(1), 156–164. https://doi.org/10.1609/hcomp.v1i1.13088

Simonyan, K., & Zisserman, A. (2015). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. https://arxiv.org/abs/1409.1556

Whitehill, J., Wu, T., Bergsma, J., Movellan, J. R., & Ruvolo, P. L. (2009). Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise. In *Advances in neural information processing systems 22* (pp. 2035–2043). Curran Associates, Inc. ISBN: 978-1-61567-911-9

Zhang, J., Sheng, V. S., Nicholson, B. A., & Wu, X. (2015). CEKA: A Tool for Mining the Wisdom of Crowds. *Journal of Machine Learning Research*, *16*(88), 2853–2858. https://jmlr.org/papers/v16/zhang15a.html

Zheng, Y., Li, G., Li, Y., Shan, C., & Cheng, R. (2017). Truth Inference in Crowdsourcing: Is the Problem Solved? *Proceedings of the VLDB Endowment*, *10*(5), 541–552. https://doi.org/10.14778/3055540.3055547