ALANNO: An Active Learning Annotation System for Mortals

Josip Jukić[♠] **Fran Jelenić**[♠] **Miroslav Bićanić** **Jan Šnajder**
University of Zagreb, Faculty of Electrical Engineering and Computing
Text Analysis and Knowledge Engineering Lab
{name.surname}@fer.hr

Abstract

Supervised machine learning has become the cornerstone of today's data-driven society, increasing the need for labeled data. However, the process of acquiring labels is often expensive and tedious. One possible remedy is to use *active learning* (AL) – a special family of machine learning algorithms designed to reduce labeling costs. Although AL has been successful in practice, a number of practical challenges hinder its effectiveness and are often overlooked in existing AL annotation tools. To address these challenges, we developed ALANNO, an open-source annotation system for NLP tasks equipped with features to make AL effective in real-world annotation projects. ALANNO facilitates annotation management in a multi-annotator setup and supports a variety of AL methods and underlying models, which are easily configurable and extensible.

1 Introduction

We are witnessing an ever-growing demand for data along with the rapid development of machine learning and deep learning algorithms. In particular, we need an abundance of labeled data to develop well-performing models, which is not easy to obtain. For many natural language processing (NLP) tasks, the labeling process, i.e., annotation, is often the most expensive and time-consuming part of developing machine learning models. The cognitive exertion of human annotators can affect their judgment, which further affects label validity. Consequently, this manifests in poor *agreement* – a proxy for label reliability, which is a prerequisite for validity (Artstein and Poesio, 2008; Paun et al., 2022). Poor label reliability and validity negatively affect the machine learning algorithm, as it is only as good as the data it consumes.

Designed to alleviate labeling issues and reduce annotation cost, *active learning* (**AL**; Settles, 2009) is a special family of machine learning algorithms. In contrast to the standard random selection of instances for labeling, a typical AL method iteratively queries the most informative instances for the underlying model to achieve the best possible performance with the fewest possible labels. AL has been shown to reduce annotation effort across machine learning applications, e.g., (Beluch et al., 2018; Zhang and Chen, 2002), especially in NLP, e.g., (Chen et al., 2012; Settles and Craven, 2008; Ein-Dor et al., 2020).

Despite the demonstrated successes of AL, many challenges are involved in deploying AL in real-world scenarios (Lowell et al., 2019; Attenberg and Provost, 2011). Unfortunately, these challenges are often overlooked in both research and practice. In particular, annotation tools that support AL rarely address the problems of unbiased evaluation of AL, imbalanced data, and stopping criteria for AL. The lack of concrete solutions for these problems hinders the effectiveness of AL. Aside from the practical challenges in AL, managing annotation campaigns is often very cumbersome, especially in multi-annotator setups (when multiple annotators are assigned to a single instance). Specifically, assigning instances to multiple annotators can be painstaking, particularly if one aims to achieve balanced combinations of annotators across instances. While there are many serviceable frameworks for simulating AL in idealized scenarios, e.g., (Danka and Horvath, 2018; Tang et al., 2019; Schröder et al., 2021), there are only a few tools for running real-world AL annotation campaigns with multiple annotators, none of them explicitly addressing the practical AL challenges.

To facilitate the creation of high-quality NLP datasets at reduced annotation costs, we developed ALANNO (**A**ctive **L**earning **Anno**tation), an open-source annotation system with AL strategies for data sampling. ALANNO's is specifically designed to address the practical challenges of AL and fa-

[♠]Equal contribution

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics
System Demonstrations, pages 228–235

cilitate the management of multi-annotator annotation projects. In particular, ALANNO guides toward more quality labels with a novel method for the balanced assignment of unlabeled instances to annotators in a multi-annotator setup. We support building gold labels by monitoring the inter-annotator agreement with task-specific metrics and agreement-aware weighted aggregation of labels. Equally important, ALANNO incorporates many features to address the major challenges of using AL in practice. Namely, we support guided learning (Attenberg and Provost, 2010) for mitigating data imbalance, and we ensure trustworthy evaluation of the underlying model on an unbiased test set and a stopping criterion to maximize the effectiveness of AL. As an essential practical solution, we enable a project-specific stopping criterion with a novel performance forecasting method based on Bayesian regression. By estimating the performance of the underlying model with hypothetically enlarged labeled sets, we enable practitioners to determine on the spot whether further annotation will only have diminishing returns. Lastly, ALANNO supports a wide range of state-of-the-art AL methods from the literature, allowing seamless inclusion of new models or methods.

In summary, our main contribution is ALANNO, an open-source AL annotation system for NLP tasks, which features (1) practical strategies for applying AL to real-world problems with a range of AL methods and (2) annotation management facilitation in a multi-annotator setup with quality control. ALANNO enables non-experts in AL to reap its benefits by accounting for key practical issues in annotation management and AL. In two case studies, we demonstrate ALANNO's two key features – balanced data assignment and AL performance forecasting. We also provide a short video[1] demonstration and release the code[2] under the Apache 2.0 license. While ALANNO has been born out of several years of experience with NLP annotations for various tasks and has evolved with each new project, it remains highly configurable, allowing easy customization and extension.

2 System Overview

We briefly describe the key aspects of ALANNO, which include projects, data assignment, label management, and annotation.

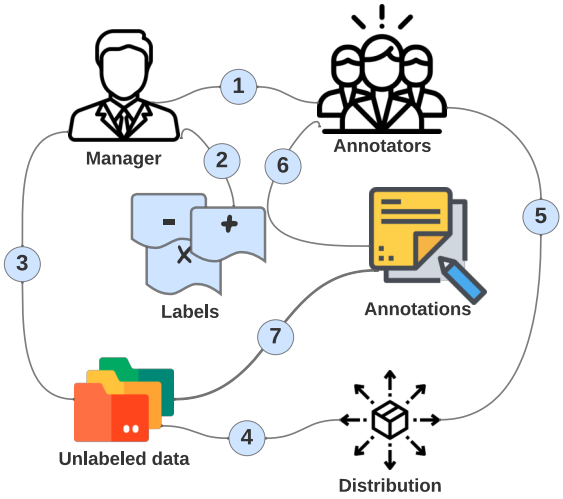

Figure 1: Organization of an ALANNO project. The lines between the icons indicate different lines of interaction, with the numbers denoting the temporal order. In brief, the project manager recruits annotators (1), creates labels (2), and imports the unlabeled data (3), which are then appointed to annotators via an assignment algorithm (4), (5). The annotators use the created labels (6) to annotate the data (7).

Projects. In ALANNO, the entire annotation process is encapsulated into a *project*, which handles the interactions between different parts of an annotation project, as depicted in Figure 1. A typical NLP annotation project is long-lasting and dynamic: annotators may be temporarily unavailable, new annotators may join an already-running project, and others may leave. ALANNO supports the managing of such a workforce dynamic. To separate the concerns and responsibilities, ALANNO defines two user roles: **project managers**, who are in charge of the annotation campaign, and **annotators**, whose task is to apply labels to the unlabeled data. At the moment, project managers can create three main types of projects: single- and multi-label classification, as well as sequence labeling tasks (e.g., named entity recognition).

Data assignment. Due to the dynamic nature of real-world annotation campaigns, it is convenient to separate the annotation process into smaller chunks. Moreover, annotation is an incremental process that often requires calibration in the initial phases. To meet these needs, the workload in ALANNO is divided into rounds, where each round can be configured independently. The project manager can specify the number of unlabeled instances to

[1] https://www.youtube.com/watch?v=hPcHPM8ttvE

[2] https://github.com/josipjukic/alanno

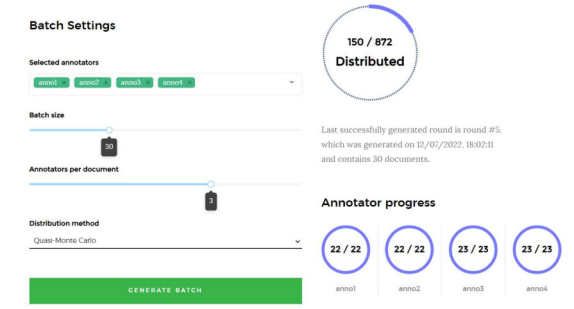

Figure 2: Data assignment interface

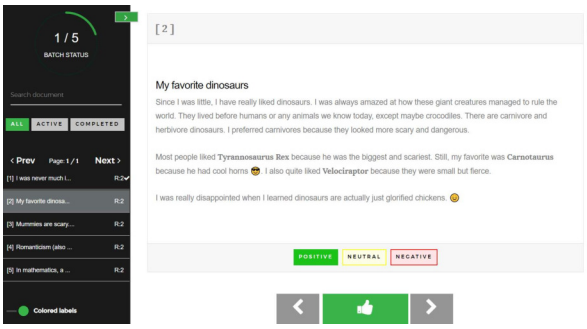

Figure 3: Annotation interface

be assigned, select annotators for the round, and the number of annotators per instance (Figure 2).

Annotation. The annotation interface (Figure 3) depends on the project type. For *classification* tasks, we support a single-label setup, where only one of the labels can be applied, and a multi-label setup with the possibility of applying multiple labels. In addition, we cover a large variety of *sequence labeling* tasks, where it is only necessary to define labels to fit the context of a specific use case. For example, one can define *organization*, *person name*, and *location* as labels for named entity recognition. Annotators can then select spans of text that fall into one of the defined categories.

Data management. The end product of an ALANNO project is the annotated dataset, which consists of labels gathered from annotators. ALANNO offers the user a choice between exporting an aggregated dataset, in which each instance appears precisely once with a single label obtained by aggregating the labels of different annotators, or a complete dataset, where each instance appears as many times as it has been annotated. The latter option is in line with recent recommendations to publish annotated datasets with the original labels rather than adjudicated labels (Kenyon-Dean et al., 2020), allowing for disagreement analysis, training of models that predict soft labels, e.g., (Pavlick and Kwiatkowski, 2019), or application of statistical label aggregation techniques, e.g., (Qing et al., 2014; Hovy and Yang, 2021; Gordon et al., 2021). Furthermore, to make it possible to follow up on earlier annotation projects, ALANNO supports partially annotated datasets where the annotations are specified as user-label pairs.

3 Features

Motivated by our experience in annotation projects and the practical challenges that emerge when deploying AL, we designed practical solutions that enable efficient labeling in real-world scenarios. We identified several key challenges, which, if not adequately addressed, may impair label quality and AL efficiency. Specifically, we focused on (1) label reliability, (2) unbiased evaluation of active learning models, (3) the stopping criterion for active learning, i.e., knowing when to terminate the active learning process, and (4) working with imbalanced data.

3.1 Annotation management

Annotation management in ALANNO is centered around the first critical challenge – label reliability. We support agreement-aware label aggregation and advanced data assignment to simultaneously promote label quality and make the management of annotation campaigns as seamless as possible.

Balanced assignment. Assigning unlabeled instances to annotators is an important aspect of the annotation process, in which combinations of annotators assigned to particular data points should be balanced to achieve more reliable labels. We have found that using uniform sampling based on pseudo-random numbers results in unbalanced combinations of annotators, with varying frequencies of different annotator tuples. To mitigate this and help improve label reliability, we developed a *Quasi-Monte Carlo* assignment method based on quasi-random numbers. In particular, we used Sobol sequences (Burhenne et al., 2011) to produce balanced combinations. In a scenario with n annotators and k annotators per data point, we draw n dimensional vectors generated from the Sobol sequence, where each element is in the $[0, 1]$ interval. We round the values to the nearest integer (0 or 1). If a particular vector has exactly k elements with value 1, we distribute the data point

to the annotators at the corresponding indices of the vector. Otherwise, we discard the vector and draw a new one. The process is guaranteed to converge since all possible combinations are covered in the first 2^n vectors from the sequence. This procedure produces balanced combinations with uniform frequencies of annotator pairs, triplets, and up to k-tuples. We demonstrate its effects in a case study in Section 4.

Monitoring agreement. In a multi-annotator setup, annotator agreement is a strong indicator of label quality. ALANNO computes the inter-annotator agreement using metrics appropriate for the particular NLP task. Specifically, we use Cohen's κ coefficient (Cohen, 1960) to evaluate pairwise agreement for binary and multi-class annotation. For the joint measure that considers all annotators simultaneously, we use Fleiss' κ (Fleiss, 1971). On the other hand, for the multi-label setup, we use Krippendorff's α coefficient (Krippendorff, 2018) paired with MASI distance (Passonneau, 2006) for both pairwise and joint agreement.

Gold labels. Aggregating labels from multiple annotators is a critical component of creating high-quality datasets. In practice, different annotators often have different reliability levels due to differences in expertise. Such differences are exceptionally prominent with large groups of annotators. Therefore, ALANNO generates *gold labels* that consider an estimate of annotators' reliability. In particular, we aggregate the labels by assigning each annotator a weight proportional to how many times they assigned the majority label to a data point (Qing et al., 2014). For tasks with multiple labels, we chose to use the majority principle. The weighted aggregation leaves room for future improvement by incorporating systems such as Multi-Annotator Competence Estimation (MACE; Hovy et al., 2013).

3.2 AL acquisition models and functions

ALANNO supports AL as one of the key features. We incorporate practical solutions to mitigate the problems of deploying AL in real-world scenarios. We first describe what the system offers in terms of *acquisition models*, i.e., the underlying models used for AL, and *acquisition functions*, i.e., AL methods.

ALANNO offers a rich palette of acquisition models for AL. We include various approaches to preprocessing tailored for a specific language for the NLP task at hand, including TF-IDF, customizable n-gram models, and word embeddings, using English as the default language. Besides English, we currently also support Croatian. ALANNO provides many traditional models, including logistic regression, SVM, and random forest classifier. We also support deep models such as recurrent networks and Transformers (Vaswani et al., 2017).

ALANNO supports a wide range of active acquisition functions for both traditional and deep learning models. Starting from uncertainty sampling (Settles, 2009), a simple but powerful family of AL methods, ALANNO covers the *least confident*, *margin*, and *entropy* methods. All uncertainty-based methods are available for single- and multi-label problems. We have also incorporated acquisition functions that focus more on data diversity, such as the *informative density* method, which leverages information about the instances in the input space and gives higher weights to instances in high-density parts of the input space. From the family of AL methods specialized for deep neural networks, ALANNO provides the *core-set* method (Sener and Savarese, 2017) and BADGE (Ash et al., 2019).

3.3 AL challenges and solutions

We describe the aforementioned practical challenges in AL (unbiased evaluation, stopping criterion, and class imbalance) and our solutions that aim to preserve AL effectiveness.

Unbiased evaluation. Before starting the annotation process, ALANNO reserves a random sample of the imported data to be used later as a test set. In each round, managers can select how many test instances drawn from the reserved pool should be labeled out of the entire batch. In this way, one can adequately evaluate the model, as the reserved pool is not affected by the sampling bias (Prabhu et al., 2019). Since the acquisition functions often rely on the acquisition model's output, it is important to decouple evaluation and AL selection. A biased test set can lead to overestimating the model's performance, establishing a vicious cycle of uninformative queries in the early stages of acquisition. This often leads to redundant labels and, consequently, poorly performing models (Attenberg and Provost, 2011).

Stopping criterion. Although several stopping criteria for active learning have been proposed (Vlachos, 2008; Zhu et al., 2010; Laws and Schütze, 2008; Bloodgood and Vijay-Shanker, 2014), they

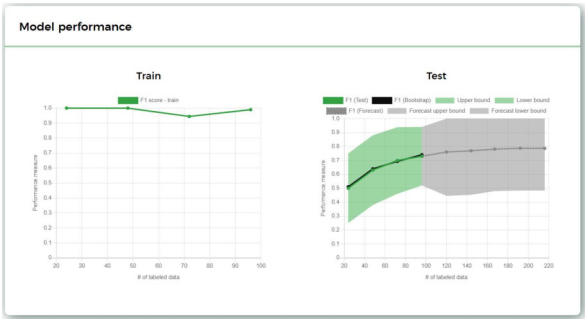

Figure 4: AL evaluation and performance forecasting. After each round, ALANNO re-trains the underlying AL model and plots the corresponding performance on the train set (the plot on the left-hand side). The performance is calculated on an unbiased test set reserved beforehand. We use F_1 score for the classification tasks, with confidence intervals approximated by bootstrapping. The plot on the right-hand side shows test performances on the data labeled so far (in green) and the forecast of performance increase with additional annotation effort (in gray).

are rarely employed in practice. We can save valuable resources by using a stopping criterion to identify when our model performs sufficiently well. Even more helpful would be the ability to forecast how much more data needs to be annotated to reach the desired model performance. To this end, we integrated a forecasting feature based on Bayesian regression, which we have implemented in Pyro (Bingham et al., 2019). The forecasting functionality is a practical solution for the stopping criterion, allowing annotation managers to gauge the trade-off between the expected boost in performance versus the additional annotation effort. Figure 4 shows an example of evaluating an active learning model and performance forecasting.

Class balancing through guided learning. AL often struggles with imbalanced data. Moreover, class balance is crucial for active learning strategies, especially in the early phases of the annotation process, as the model may have difficulty learning classes with low frequency. To address this, ALANNO supports *guided learning*, also known as *active search* (Attenberg and Provost, 2010). The main idea of guided learning in NLP is to use keywords to search for data points in the minority class. This way, users can annotate the retrieved data to make the class frequency distribution more uniform. We use BM25 (Robertson et al., 2009) as the retrieval algorithm for guided learning.

	$k=2$	$k=3$	$k=4$	$k=5$
UNIFORM	132.66	52.14	18.09	3.38
QMC	0.75	0.42	0.15	.03

Table 1: The average variance of k-tuple frequencies. UNIFORM denotes the standard uniform sampling of annotators, while QMC stands for our Quasi-Monte Carlo assignment method. We simulated the assignment of 1,000 unlabeled instances with ten annotators in total and five annotators per instance. We report the average variance of frequencies across 1,000 runs.

4 Case Studies

In the following case studies, we highlight the two essential features of ALANNO, namely balanced data assignment and AL performance forecasting.

Case study 1: Balanced assignment. To compare our Quasi-Monte Carlo assignment method with uniform annotators combinations, we ran simulations of distributing unlabeled instances to annotators. As Table 1 shows, our method achieves more balanced combinations compared to the standard uniform sampling, ranging from pairs and up to k-tuples, where k is the number of annotators per data point.

Case study 2: AL performance forecasting. To demonstrate the forecasting feature in ALANNO, we conducted a case study on the Stanford Sentiment Treebank (**SST;** Socher et al., 2013) and subjectivity (**SUBJ;** Pang and Lee, 2004) datasets. We used a simple logistic regression model with TF-IDF vectors and *least confident* sampling method for SST and BERT with BADGE sampling method for SUBJ. We then compared random sampling to AL and used our forecasting technique to predict the performance of AL. In each step, we sampled 200 data points for SST and 50 data points for SUBJ from the pool of unlabeled data, simulating the annotation process. We re-trained the models in each AL step and evaluated them on the test set. Figure 5 demonstrates the usefulness of performance forecasting, which provides a possibility to decide on the trade-off between the additional annotation cost and the expected increase in performance.

5 Related Work

As the popularity of machine learning and deep learning grows, so does the need for annotated data. Since high-quality data is imperative for high-quality machine learning models, data annotation

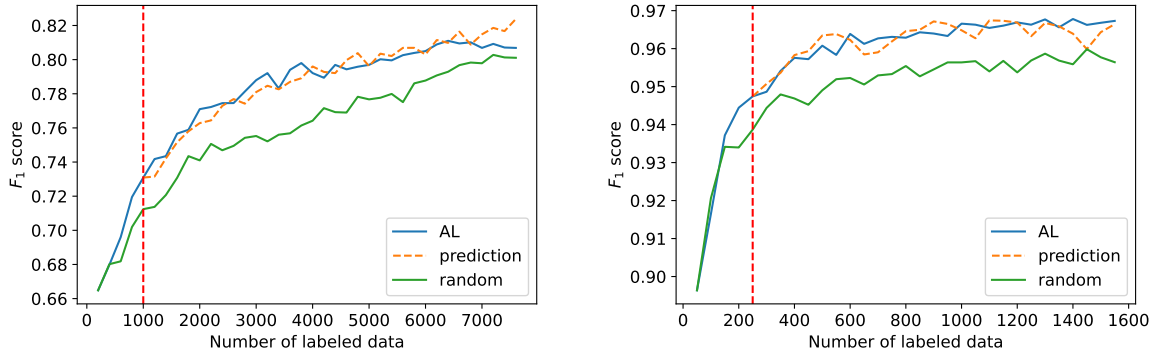

Figure 5: AL performance forecasting. The subfigures show the predictions of F_1 score gain with a hypothetical increase in the number of labeled data. The green line pertains to the random selection baseline, the blue one is the corresponding AL selection, and the dashed orange line is the forecast of F_1 score for AL selection. The dashed red line represents the boundary between the steps the forecast model was trained on (left of the line) and the steps whose values the forecast model was predicting (right of the line). The left subfigure shows the results for SST with logistic regression using uncertainty sampling as the AL method, while the right subfigure corresponds to SUBJ with BERT using BADGE for AL sampling.

has become a lucrative industry, and new tools are constantly emerging. There are commercial tools such as Prodigy,[3] V7,[4] and Hasty.[5] However, these tools hide their full functionality behind a paywall. In contrast to the mentioned commercial system, several open-source annotation tools have appeared recently, such as Label Sleuth (Shnarch et al., 2022), Label Studio,[6], INCEpTION (Klie et al., 2018), MATILDA (Cucurnia et al., 2021), and Paladin (Nghiem et al., 2021).

Label Sleuth is an elegant annotation system designed to make NLP accessible for non-experts. The system enables AL selection for labeling. However, it only supports simple binary classification with a single annotator per project.

Label Studio is offered as an open-source system and a paid enterprise version. While the paid version supports active learning, the free version is limited to random selection. The system supports multiple annotators but with minimal functionalities in managing the annotations.

INCEpTION is a highly configurable tool that supports AL and multi-annotator setups. However, the system is hard to use, as it requires external libraries to integrate a model for AL purposes.

MATILDA is a platform for dialogue annotation in a multi-annotator setup with support for multiple languages.

Paladin integrates active learning and supports multi-label classification.

To the best of our knowledge, ALANNO is the only AL annotation tool that explicitly addresses practical challenges in AL. ALANNO also differs from the above-mentioned systems in implementing practical solutions for managing multi-annotator annotation projects.

6 Conclusion

ALANNO is an open-source annotation system for natural language processing tasks powered by active learning. The system addresses the critical practical challenges of active learning in real-world annotation projects that have previously been overlooked. ALANNO enables non-experts in active learning to conduct effective annotation campaigns by supporting solutions for unbiased evaluation, stopping criterion for active learning, and class balancing. Additionally, the system facilitates annotation management in a multi-annotator setup, emphasizing label quality through agreement monitoring, agreement-aware label aggregation, and a novel method for the balanced assignment of unlabeled instances to annotators.

Acknowledgements

We thank the reviewers for their comments and interesting suggestions for future improvements. We also thank the people from TakeLab (Text Analysis and Knowledge Engineering Lab) who helped us in developing the system over the years.

[3] https://prodi.gy/
[4] https://www.v7labs.com/
[5] https://hasty.ai/
[6] https://labelstud.io/

References

Ron Artstein and Massimo Poesio. 2008. Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4):555–596.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2019. Deep batch active learning by diverse, uncertain gradient lower bounds. *CoRR*, abs/1906.03671.

Josh Attenberg and Foster Provost. 2010. Why label when you can search? Alternatives to active learning for applying human resources to build classification models under extreme class imbalance. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 423–432.

Josh Attenberg and Foster Provost. 2011. Inactive learning? Difficulties employing active learning in practice. *ACM SIGKDD Explorations Newsletter*, 12(2):36–41.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. 2018. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9368–9377.

Eli Bingham, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman. 2019. Pyro: Deep universal probabilistic programming. *The Journal of Machine Learning Research*, 20(1):973–978.

Michael Bloodgood and K Vijay-Shanker. 2014. A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. *arXiv preprint arXiv:1409.5165*.

Sebastian Burhenne, Dirk Jacob, and Gregor Henze. 2011. Sampling based on Sobol' sequences for Monte Carlo techniques applied to building simulations. pages 1816–1823.

Yukun Chen, Subramani Mani, and Hua Xu. 2012. Applying active learning to assertion classification of concepts in clinical text. *Journal of biomedical informatics*, 45(2):265–272.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Davide Cucurnia, Nikolai Rozanov, Irene Sucameli, Augusto Ciuffoletti, and Maria Simi. 2021. MATILDA - multi-AnnoTator multi-language InteractiveLightweight dialogue annotator. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 32–39, Online. Association for Computational Linguistics.

Tivadar Danka and Peter Horvath. 2018. modal: A modular active learning framework for Python.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Mitchell L Gordon, Kaitlyn Zhou, Kayur Patel, Tatsunori Hashimoto, and Michael S Bernstein. 2021. The disagreement deconvolution: Bringing machine learning performance metrics in line with reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130, Atlanta, Georgia. Association for Computational Linguistics.

Dirk Hovy and Diyi Yang. 2021. The importance of modeling social factors of language: Theory and practice. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 588–602, Online. Association for Computational Linguistics.

Kian Kenyon-Dean, Edward Newell, and Jackie Chi Kit Cheung. 2020. Deconstructing word embedding algorithms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8479–8484, Online. Association for Computational Linguistics.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico. Association for Computational Linguistics.

Klaus Krippendorff. 2018. *Content analysis: An introduction to its methodology*. Sage publications.

Florian Laws and Hinrich Schütze. 2008. Stopping criteria for active learning of named entity recognition. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 465–472.

David Lowell, Zachary C. Lipton, and Byron C. Wallace. 2019. Practical obstacles to deploying active learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*

and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 21–30, Hong Kong, China. Association for Computational Linguistics.

Minh-Quoc Nghiem, Paul Baylis, and Sophia Ananiadou. 2021. Paladin: an annotation tool based on active and proactive learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 238–243, Online. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Rebecca Passonneau. 2006. Measuring agreement on set-valued items (MASI) for semantic and pragmatic annotation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Silviu Paun, Ron Artstein, and Massimo Poesio. 2022. Statistical methods for annotation analysis. *Synthesis Lectures on Human Language Technologies*, 15(1):1–217.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Ameya Prabhu, Charles Dognin, and Maneesh Singh. 2019. Sampling bias in deep active classification: An empirical study. *arXiv preprint arXiv:1909.09389*.

Ciyang Qing, Ulle Endriss, Raquel Fernández, and Justin Kruger. 2014. Empirical analysis of aggregation methods for collective annotation.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Christopher Schröder, Lydia Müller, Andreas Niekler, and Martin Potthast. 2021. Small-text: Active learning for text classification in python. *CoRR*, abs/2107.10314.

Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Burr Settles. 2009. Active learning literature survey. Computer sciences technical report.

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

Eyal Shnarch, Alon Halfon, Ariel Gera, Marina Danilevsky, Yannis Katsis, Leshem Choshen, Martin Santillan Cooper, Dina Epelboim, Zheng Zhang, Dakuo Wang, Lucy Yip, Liat Ein-Dor, Lena Dankin, Ilya Shnayderman, Ranit Aharonov, Yunyao Li, Naftali Liberman, Philip Levin Slesarev, Gwilym Newton, Shila Ofek-Koifman, Noam Slonim, and Yoav Katz. 2022. Label Sleuth: From unlabeled text to a classifier in a few hours.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria. Association for Computational Linguistics.

Ying-Peng Tang, Guo-Xiang Li, and Sheng-Jun Huang. 2019. Alipy: Active learning in Python.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Andreas Vlachos. 2008. A stopping criterion for active learning. *Computer Speech & Language*, 22(3):295–312.

Cha Zhang and Tsuhan Chen. 2002. An active learning framework for content-based information retrieval. *IEEE transactions on multimedia*, 4(2):260–268.

Jingbo Zhu, Huizhen Wang, Eduard Hovy, and Matthew Ma. 2010. Confidence-based stopping criteria for active learning for data annotation. *ACM Transactions on Speech and Language Processing (TSLP)*, 6(3):1–24.