ALAMBIC : Active Learning Automation with Methods to Battle Inefficient Curation

Charlotte Nachtegael
Machine Learning Group, ULB
IBsquare, ULB-VUB
charlotte.nachtegael@ulb.be

Jacopo De Stefani
TPM-ESS-ICT, TUDelft
Machine Learning Group, ULB
J.deStefani@tudelft.nl

Tom Lenaerts
Machine Learning Group, ULB
IBsquare, ULB-VUB
AI Lab, VUB
Tom.Lenaerts@ulb.be

Abstract

In this paper, we present ALAMBIC, an open-source dockerized web-based platform for annotating text data through active learning for classification tasks. Active learning is known to reduce the need of labelling, a time-consuming task, by selecting the most informative instances among the unlabelled instances, reaching an optimal accuracy faster than by just randomly labelling data. ALAMBIC integrates all the steps from data import to customization of the (active) learning process and annotation of the data, with indications of the progress of the trained model that can be downloaded and used in downstream tasks. Its architecture also allows the easy integration of other types of models, features and active learning strategies. The code is available on https://trusted-ai-labs.github/ALAMBIC/ and a video demonstration is available on https://youtu.be/4oh8UADfEmY.

1 Introduction

Data annotation is crucial for any machine learning task. Datasets of high-quality are obtained through manual labelling which requires both considerable time and sometimes also expertise. Active learning aims to reduce the need for labelled data by starting from a partially labelled dataset and gradually selecting the most informative instances among the unlabelled instances (Settles, 2009; Baldridge and Palmer, 2009). This incremental training of a machine learning model can thus actively select the instances it finds to be the most informative, asking the person involved to provide a true label for the selected data. Several strategies exist to define informativeness, such as those based on uncertainty of the predictions obtained for unlabelled instances (Settles, 2009) or based on the difference of their features with respect to the training set (Sener and Savarese, 2018). The model is then trained with the newly obtained labelled set and can again select instances to be labeled. This process is repeated up until a specific criterion is reached, such as a desired accuracy or a maximum number of instances labelled.

We present in this work **ALAMBIC** (**A**ctive **L**earning **A**utomation with **M**ethods to **B**attle **I**nefficient **C**uration), an open-source dockerized web-based platform for active-learning-based text classification, allowing for a full customisation of the active learning process, from the choice of the model, its features and parameters, and the active learning strategy[1]. It allows the study of the usefulness of active learning on a given labelled dataset and model, as well as the annotation of text instances with active learning.

2 Related works

Several active learning libraries have been developed, such as ALiPy (Tang et al., 2019), modAL (Danka and Horvath), scikit-activeml (Kottke et al., 2021), for traditional machine learning methods and DISTIL (Beck et al., 2021), SmallText (Schröder et al., 2021), the low-resource Text classification framework (Ein-Dor et al., 2020) or the ALToolbox (Tsvigun et al., 2022) which also contains active learning methods for deep learning. However, these libraries do not provide user interface or if they do, they require the user to be reasonably skilled in coding. Tools such as Prodigy [2], APLenty (Nghiem and Ananiadou, 2018), AlpacaTag (Lin et al., 2019), Paladin (Nghiem et al.,

[1]Code source available on https://github.com/Trusted-AI-Labs/ALAMBIC

[2]https://prodi.gy/

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics
System Demonstrations, pages 117–127

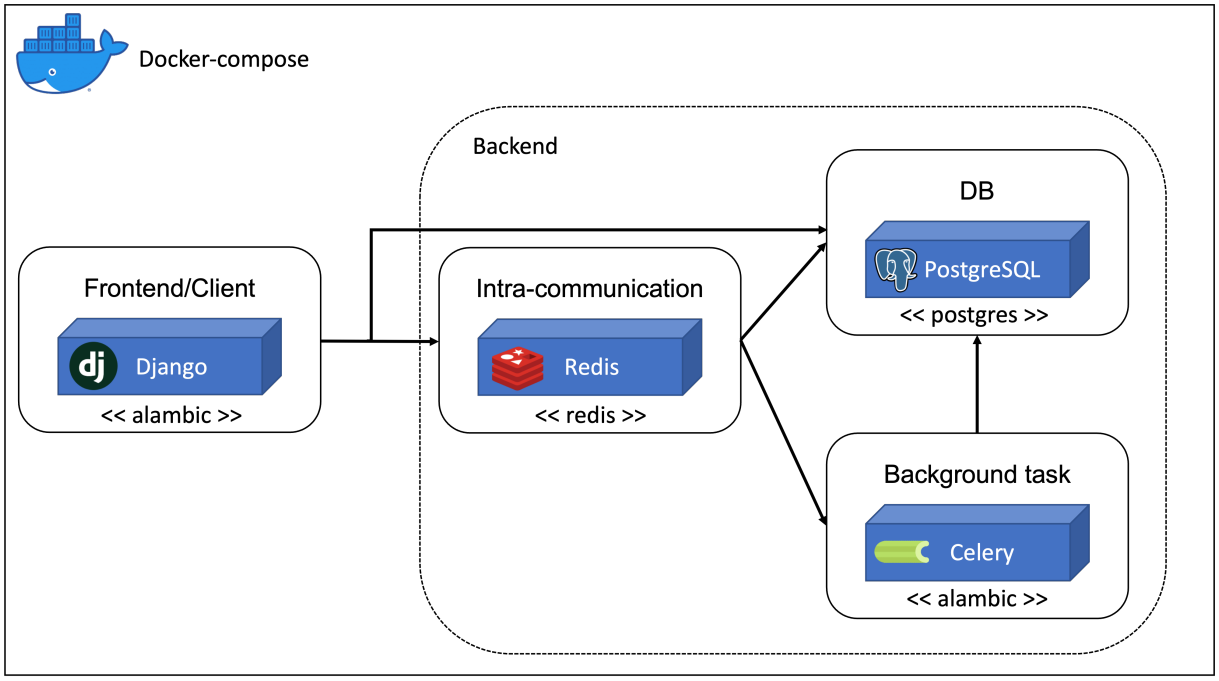

Figure 1: Diagram of the orchestrated deployment for ALAMBIC. Each box with a full line corresponds to a container. The blue box inside the container is the main service of the container. The name of the docker image for each container is indicated below the blue box.

2021), or Label Sleuth (Shnarch et al., 2022) tried to fill the gap by offering a user interface for the annotation process combined with an active learning setting. While these tools offer a friendly user-interface to annotate text classification or sequence labelling tasks, they either require a high-to-expert knowledge in programming languages, are not open source or are limited in the choice of the parameters in the active learning process, such as the model or the selection strategy.

At the time of writing, and to the best of our knowledge, **ALAMBIC** is the first free annotation tool for text classification with a user-friendly interface allowing a complete control of the user on the parameters of the features, model and active learning strategy, with no coding skills needed. Moreover, in addition to support the annotation task, it also allows the study of different active learning strategies with chosen features and models to evaluate the best strategy within that specific context. Its dockerized form also allows for an easy deployment and usage on diverse platforms.

3 Design and implementation

ALAMBIC is a web-based platform built in Python using Django framework [3], combined with Celery [4], Redis [5] and PostgreSQL [6] in a Docker Compose setting (Fig. 1). The installation requires only few command lines and is quite easy, even for people not familiar with Docker or GitHub (see Appendix A). The framework is divided in five main interfaces : (i) the import data interface, (ii) the setup and customisation of the (active) learning process, (iii) the progress of the active learning process, (iv) annotation interface and (v) the result interface. Each of these interfaces will be described below in detail[7].

3.1 Data import and task choice

ALAMBIC supports the import of the data in raw text with a reference file containing the path of the files and their labels if already available. The supported format is a TSV file with the path of the text files and their optional labels.

The user can also choose the annotation task. At the moment only multi-class classification is implemented.

Future developments will include annotation for relation extraction, as well as additional formats, such as JSON and XML, for import.

[3] https://www.djangoproject.com/

[4] https://github.com/celery/celery

[5] https://redis.io/

[6] https://www.postgresql.org/

[7] Full documentation is available at https://trusted-ai-labs.github.io/ALAMBIC/

If several files are provided, the user can choose among them through the dedicated user interface. Once imported, the data is accessible through the Django admin interface.

3.2 Setup of the learning process

Once the data has been added to the database, the user can setup each part of the learning process, from the model, its parameters and the features used as inputs.

The user can then choose to either analyse the impact of different active learning strategies in order to evaluate which one would be the most suitable with their data and choice of learning process, or use a specific active learning strategy to annotate the unlabeled part of the dataset. In the first case, the user is invited to choose the number of cross-validation folds, number of repetitions and size of the initial training seed, as well as which strategies they want to evaluate, ensuring a robust analysis of the process. In the second case, the user can choose the portion of the dataset to use as test set to evaluate the performance of the model at each iteration of the active learning process, the size of the training seed, the active learning strategy and the stop criterion, which will stop the active process once a specific criteria is met, such as a minimum accuracy, or a number of annotations done.

Model and feature implementations are based on scikit-learn (Pedregosa et al., 2011). While only a fraction of these are currently implemented, the architecture allows for an easy extension to functionalities already implemented in scikit-learn or having a similar programming interface. Our documentation offers a dedicated section for people more at ease in coding with Python to guide them through the integration of new models or features.

Active learning strategies are implemented from the ALiPy library (Tang et al., 2019). We choose to use only the strategies using the trained model in a pool-based scenario, i.e. all the unlabelled instances are considered for labelling at each iteration step.

3.3 Analysis and annotation interface

Once either type of processes, i.e. study of different active learning strategies or annotation, is launched, the process can be followed on an interface, displaying a plot of the performance of the model up until the current iteration.

All the plots are interactive, meaning that the user can zoom to observe a specific range of data, highlight and make disappear specific observations as to not crowd the graphic.

3.3.1 Study of different active learning strategies

In this case, the interface shows the performance in terms of accuracy of each of the strategies while the model goes through the different cross-validation folds and repetitions. The entire process is automated and can be followed in real-time.

Once the process is finished, the user can download the performances of the model generated during the whole analysis. Performances measures currently include accuracy, precision, recall, F-score and Matthews correlation coefficient.

3.3.2 Annotation using active learning

During the training of the model and the selection of the instances to label, the user can also observe in real time the accuracy of the model with the currently labelled data used as a training set.

Once instances have been selected to be labelled, the user is brought to a page where the text to label is displayed (Fig 2). They can either select an existing label or create a new one in the below dropdown menu. Above the text, a interactive plot with all the performance measures across the iterations is shown. This should have a positive impact on the motivation of the annotator by showing the effect of their work directly during the learning process.

3.4 Results interface

Once the process finished, the user can download for both types of processes the performances visualized in the plots. If ALAMBIC is used for annotating a dataset, the annotated dataset and the trained model during the last iteration can also be downloaded.

The performances are available in a CSV format with the different performance measures for each iteration, and optionally the repetitions, of the active learning process.

The annotated dataset format is a CSV file with the path of the text file, the ground-truth label if available, the manual label given during the antation process, the label predicted by the model trained with ALAMBIC, as well as if the instance was part of the training set, the test set or none of them.

The model is exported in a compressed joblib format, compatible with most of the machine learning libraries for downstream import and usage.

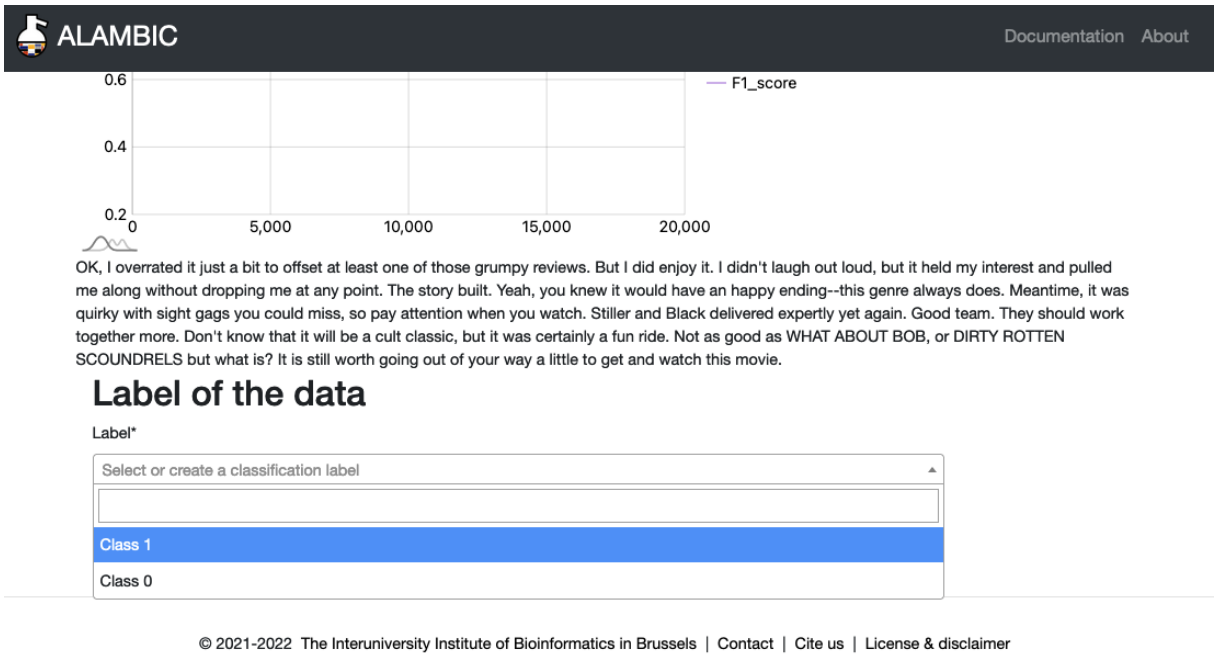

Figure 2: Annotation interface for text classification in ALAMBIC. The text to label is displayed below an interactive plot showing the performances of the model trained on the labelled data across the active learning iterations. The dropdown menu below the text to annotate allows the selection of an existing label or the creation of a new one by typing the new label in the text box.

4 Use Cases

The platform supposes that the user has a partially labelled dataset, a preferred model and features to use, and wants to label a portion of the remaining unlabelled dataset in a way that this portion will be the most informative to obtain a high-performance model. ALAMBIC can be used for two main use cases :

- **Comparison of active learning strategies:** ALAMBIC can be used to compare the performance of different active learning strategies, to determine which strategy to choose under given parameters/conditions.
- **Annotation of datasets:** ALAMBIC allows to choose one of the implemented active learning strategies and proceed with the annotation. The resulting annotations, performances measures and trained model can be exported and used for downstream analysis or automatic annotation.

5 Experiments and Results

We used the training set of the *Large Movie Review Dataset* (Maas et al., 2011), consisting in 25,000 highly polarized movie reviews, 12,500 positive and 12,500 negative.

For the analysis, we selected randomly 500 instances from each class.

Using ALAMBIC, we tested several active learning methods, including an uncertainty-based strategy (Settles, 2009), i.e. strategy based on different measures of uncertainty computed with the prediction output of the model; Core-set (Sener and Savarese, 2018), a selection method which tries to select the most different from the current training set and representative subset of unlabelled instances; and of course the random sampling as a baseline.

Each active learning method was tested for five folds of cross-validation, with three repetitions of the active learning process for the same test set with different initial training seeds (of a size of 10% of the remaining dataset). This means that 15 experiments were conducted for each strategy. Resulting performances are averaged for each iteration step. For each step, 50 instances are selected to be added.

The experiment was conducted with a Support Vector Machine (SVM) and a Random Forest (RF), with the default parameters proposed in their im-

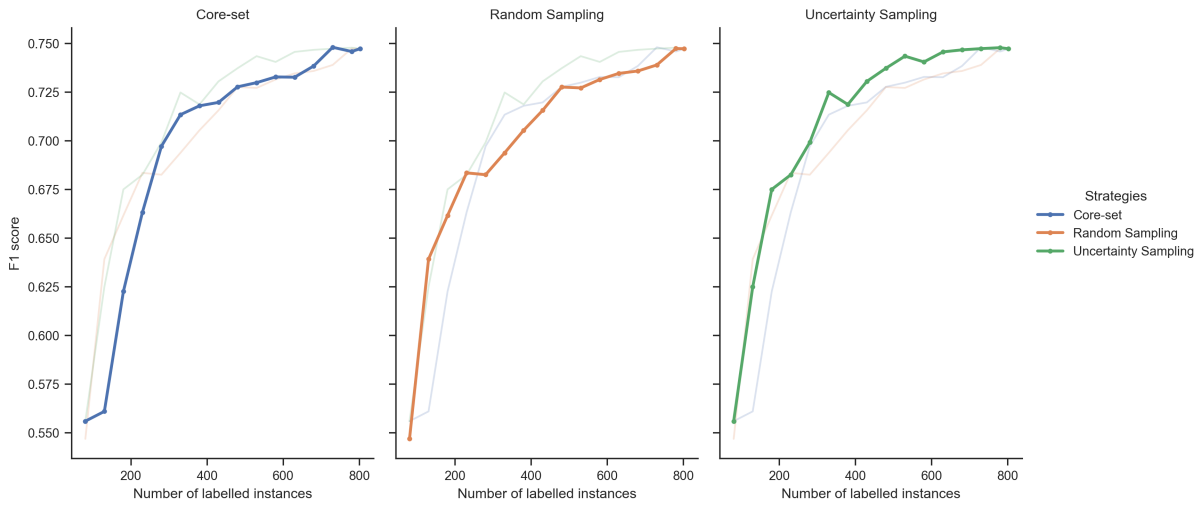

Figure 3: F-score obtained during the analysis of several active learning strategies with the SVM for a subset of 1000 samples of the Large Movie Reviews Dataset. Each panel highlights the results obtained for one specific strategy, with the other strategies greyed out.

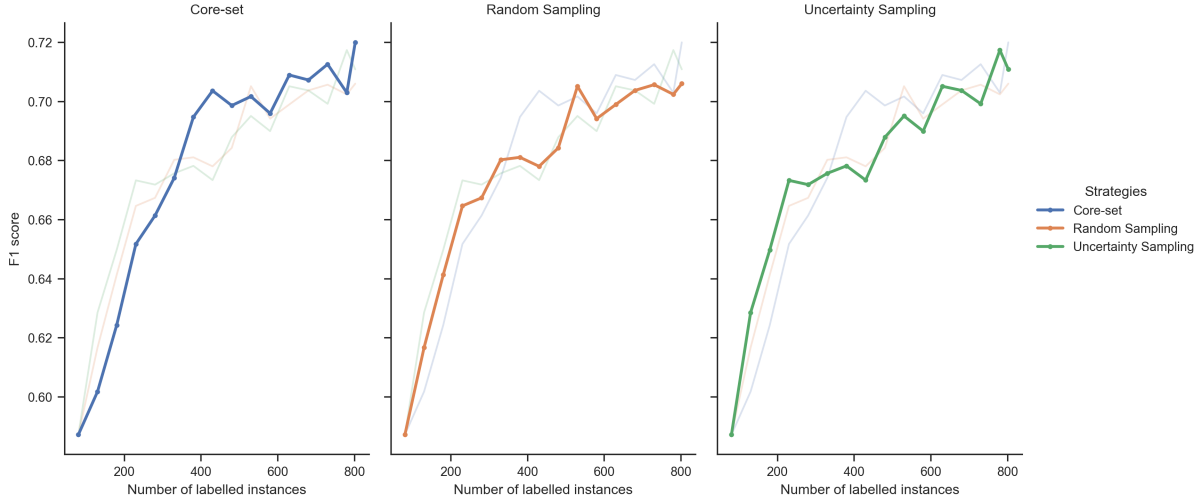

Figure 4: F-score obtained during the analysis of several active learning strategies with the RF for a subset of 1000 samples of the Large Movie Reviews Dataset. Each panel highlights the results obtained for one specific strategy, with the other strategies greyed out.

plementation in scikit-learn. The text was first pre-processed by ignoring the stop words, then the Term Frequency -Inverse Document Frequency (TF-IDF) was computed for each term, with a minimum document frequency of 0.1 and a maximum of 0.9. Only the top 3000 features were kept to build the vocabulary, ordered by term frequency across the corpus.

Figure 3 and 4 display the F-scores obtained for the subset of 1000 samples. We could observe that it would be preferable to use the uncertainty sampling method coupled with the SVM, as we see that while it performs less efficiently in the first few iterations, it clearly performs better and reached an optimal performance earlier than the random sampling or core-set method. However, if one wanted to use a RF, then it would be better to use the core-set method, as while it does not perform well in the first iterations, it outperforms the outer two selection methods. Moreover, for the RF, the uncertainty method has a similar performance as the random sampling.

Those observations highlight the importance of such analysis before using active learning in a real setting in order to choose the most optimal strategy. More advanced cases will be explored outside this paper.

6 Conclusion and Future Work

We presented ALAMBIC, an open-source web-based platform for the study and use of active learning in text classification. It can help people with low-to-no coding skills to use in the most efficient way active learning methods to build high-quality NLP datasets at a faster pace. First, they can compare with their choice of parameters and models different active learning strategies and then annotate their data with the strategy obtaining the best performance.

Further developments of the tools include the addition of other NLP annotation tasks, the integration of deep learning models and their related active learning methods. While the later development would reduce the ease of deployment for non-experts in coding, this would offer more state-of-the-art models for further automatic annotation.

Limitations

At the moment, only traditional machine learning methods can be used with our framework. The expansion to deep learning methods would bring models with better accuracy and make disappear the need to study also different features.

Using active learning in practice is also subject to several limitations. First, finding the right initial pool for an active learning setting will have an important impact on the overall performance of the active learning process. Moreover, the choice of the evaluation set is difficult as 1) it has to be representative of a dataset whom distribution could be unknown and 2) large enough to evaluate the performances of the model (such as recall, notably sensitive to class distribution). The latter aspect in particular would be in conflict with the spirit of active learning which tries to limit the annotation as much as possible. Finally, one has to keep in mind that the oracle/annotator is not always right and could in consequence introduce noise in the active learning process. While some strategies could be implemented to fight those practical issues (Yang and Loog, 2022; Paul et al., 2020), they were not implemented or taken into account in our work.

While it only supports few models and features at the moment, it can be easily extended to any models and features developed by scikit-learn.

ALAMBIC has only been tested and implemented for the English language. Spacy (Honnibal et al., 2020), which is used for some pre-processing steps, can be however adapted for many other languages.

The platform has also not been developed for a multi-annotators or crowd annotator contexts, which means that only one annotator can work at a time on the annotation task.

Acknowledgements

Basic architecture and design was largely inspired by the work done by Alexandre Renaux for ORVAL[8] (Renaux et al., 2019). This work was supported by Service Public de Wallonie Recherche under grant n° 2010235 - ARIAC by DIGITALWALLONIA4.AI. We would also like to thank the anonymous reviewers for their helpful comments.

References

Jason Baldridge and Alexis Palmer. 2009. How well does active learning *actually* work? Time-based evaluation of cost-reduction strategies for language documentation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 296–305, Singapore. Association for Computational Linguistics.

Nathan Beck, Durga Sivasubramanian, Apurva Dani, Ganesh Ramakrishnan, and Rishabh Iyer. 2021. Effective evaluation of deep active learning on image classification tasks.

Tivadar Danka and Peter Horvath. modAL: A modular active learning framework for Python. Available on arXiv at `https://arxiv.org/abs/1805.00979`.

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.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Daniel Kottke, Marek Herde, Tuan Pham Minh, Alexander Benz, Pascal Mergard, Atal Roghman, Christoph Sandrock, and Bernhard Sick. 2021. scikitactiveml: A library and toolbox for active learning algorithms. *Preprints*.

Bill Yuchen Lin, Dong-Ho Lee, Frank F. Xu, Ouyu Lan, and Xiang Ren. 2019. AlpacaTag: An active learning-based crowd annotation framework for sequence tagging. In *Proceedings of the 57th Annual Meeting of*

[8] `https://orval.ibsquare.be`

the Association for Computational Linguistics: System Demonstrations, pages 58–63, Florence, Italy. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Minh-Quoc Nghiem and Sophia Ananiadou. 2018. APLenty: annotation tool for creating high-quality datasets using active and proactive learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 108–113, Brussels, Belgium. 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.

Sudipta Paul, Shivkumar Chandrasekaran, B. S. Manjunath, and Amit K. Roy-Chowdhury. 2020. Exploiting context for robustness to label noise in active learning. *ArXiv*, abs/2010.09066.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Alexandre Renaux, Sofia Papadimitriou, Nassim Versbraegen, Charlotte Nachtegael, Simon Boutry, Ann Nowé, Guillaume Smits, and Tom Lenaerts. 2019. ORVAL: a novel platform for the prediction and exploration of disease-causing oligogenic variant combinations. *Nucleic Acids Research*, 47(W1):W93–W98.

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

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin–Madison.

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. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics.

Ying-Peng Tang, Guo-Xiang Li, and Sheng-Jun Huang. 2019. ALiPy: Active learning in python. Technical report, Nanjing University of Aeronautics and Astronautics. Available as arXiv preprint https://arxiv.org/abs/1901.03802.

Akim Tsvigun, Leonid Sanochkin, Daniil Larionov, Gleb Kuzmin, Artem Vazhentsev, Ivan Lazichny, Nikita Khromov, Danil Kireev, Aleksandr Rubashevskii, Alexander Panchenko, Olga Shahmatova, Dmitry Dylov, Igor Galitskiy, and Artem Shelmanov. 2022. ALToolbox: A set of tools for active learning annotation of natural language texts. In *Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 406–434, Abu Dhabi, UAE. Association for Computational Linguistics.

Yazhou Yang and Marco Loog. 2022. To actively initialize active learning. *Pattern Recognition*, 131:108836.

Ethics Statement

We have considered eventual ethical impacts of our developed tool and evaluated the following questions :

- **Does the paper describe how the technology would be deployed in actual use cases?** Yes, our tool is dockerized, thusly easily deployable. We give further details on the deployment in our documentation[9]. Our aim is to offer a tool which is easy to use for non-coding experts.
- **Does the task carried out by the computer match how it would be deployed?** Yes, it is exactly as described in the paper, as we created a tool directly destined for users for annotation purposes.
- **Does the paper address possible harms when the technology is being used as intended and functioning correctly?** The tool can only be used in a local setting for an optimised, faster annotation. No harm can be directly induced by the tool itself. However, the users could use the tool to annotate in an harmful way data and maliciously spread the dataset, with biases and false information.

[9] https://trusted-ai-labs.github.io/ALAMBIC/

- **Does the paper address possible harms when the technology is being used as intended but giving incorrect results?** As the task concerns annotation, the only harm brought by the tool would come from the inefficient selection of instances to be labelled, which would only impact the performance of the labelling, but not bring harm directly to the human user.

- **Does the paper address possible harms following from potential misuse of the technology?** It highly depends on which type of data the user wants to annotate. The misuse would come from the data content and what will do the user with this data, such as spreading wrongly annotated datasets.

- **If the system learns from user input once deployed, does the paper describe checks and limitations to the learning?** The trained model learns from the labelled dataset, which is expanded by the user during the annotation process. However, the learning process is limited to the annotation process or the analysis process.

A Installation commands

Docker, Docker compose and git needs to be installed. The commands to install the tool and launch everything can be resumed in :

- Clone the repository
- Build the docker

Listing 1: Commands to install ALAMBIC

```
git clone https://github.com/Trusted-AI-Labs/ALAMBIC.git
cd ALAMBIC/
docker-compose up
```

B Interfaces examples

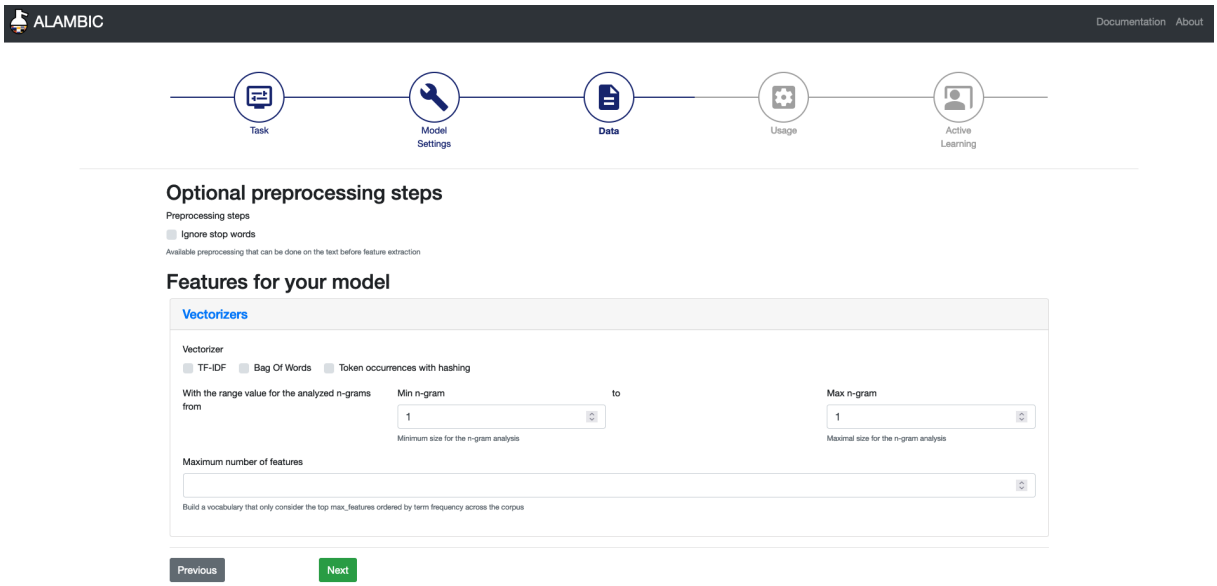

Figure 5: Choice of the model inputs/features.

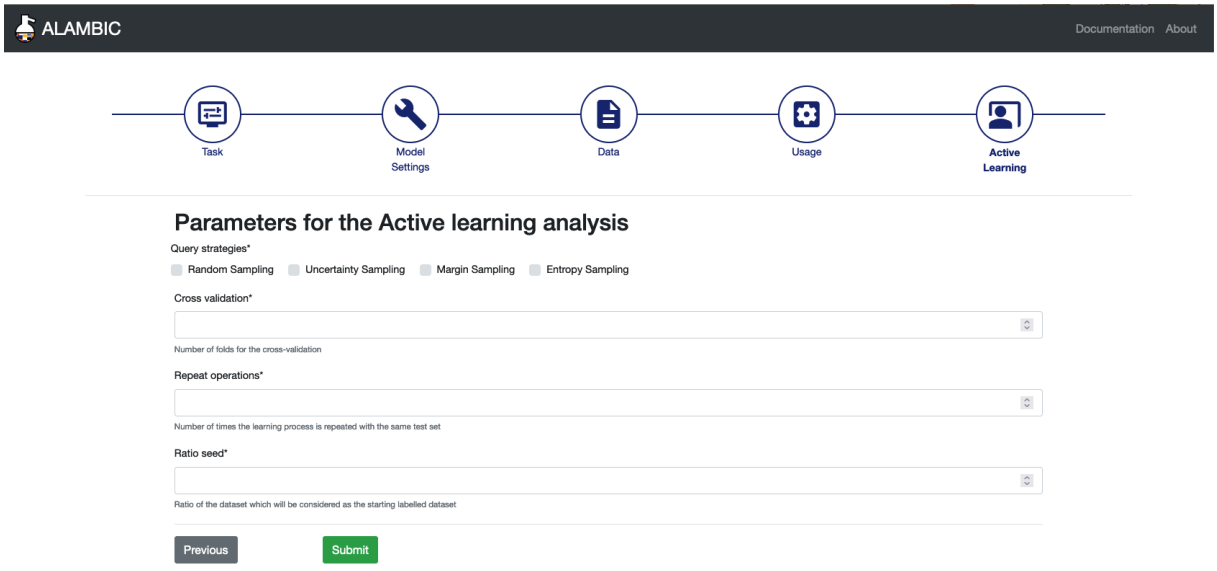

Figure 6: Parameters choice for an analysis of the use of different active learning strategies. Specific parameters include which strategies to evaluated, the number of cross-validations, number of repeats with different training seed and the ratio of the labelled set to be used as training seed.

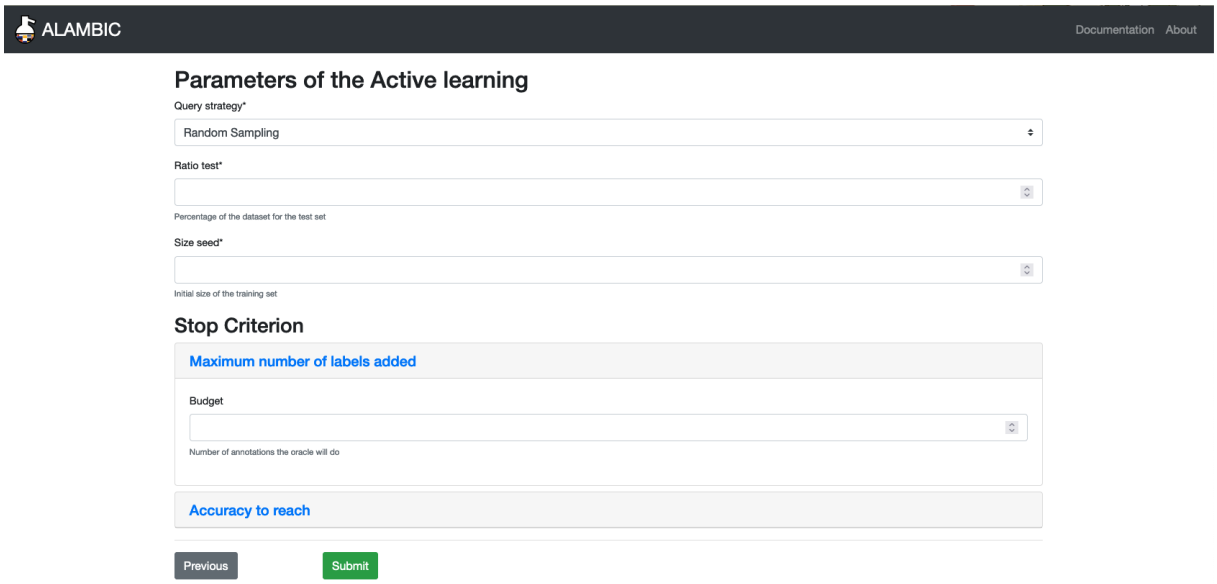

Figure 7: Parameters choice for the annotation using active learning. Specific parameters include the strategy to use, the ratio of the dataset to use as a test set, the size of the training seed and the stop criterion of the active learning process.

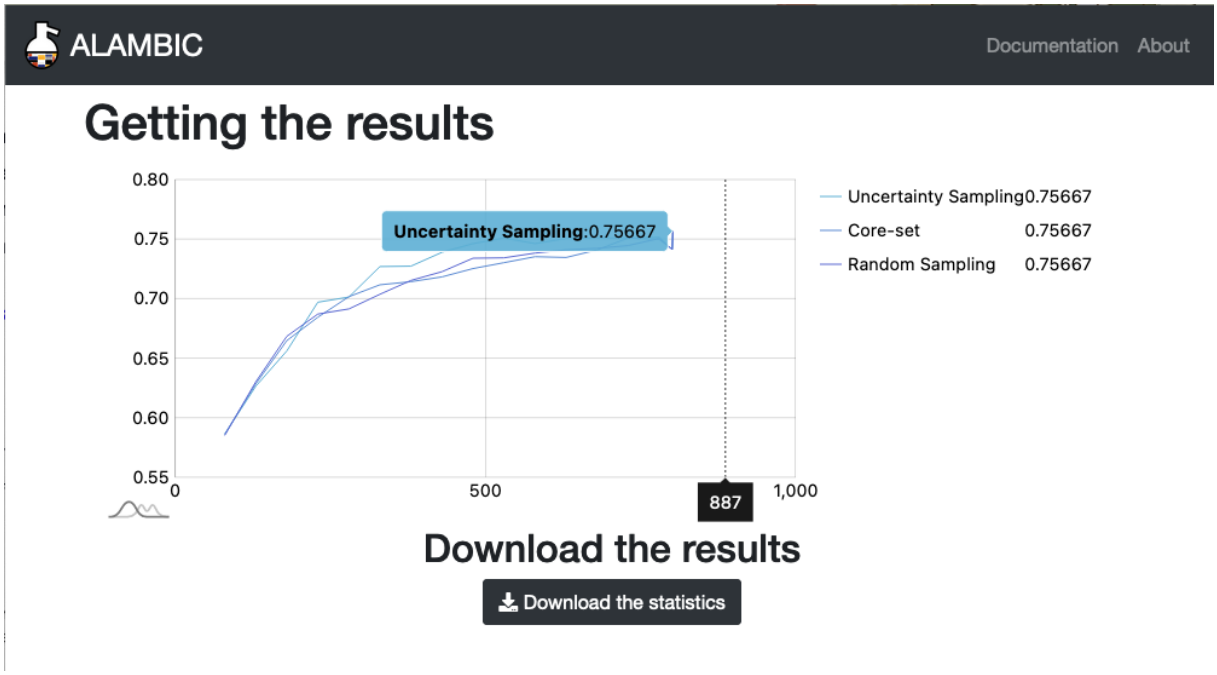

Figure 8: Example of result for the analysis of different active learning strategies.