Improving Text Classifier Performance through Human-in-the-Loop Error Correction: Enhancing Learning from Explanations

Abstract

Text classification models use natural language processing to analyse text and assign labels. In crisis scenarios such as floods and earthquakes, text classifiers can identify and forward emergency text reports from social media to relevant agencies. However, the effectiveness of text classifiers relies heavily on a large amount of labelled training data, which can be scarce and difficult to obtain[1]. In addition, training large amounts of labelled data can delay model response times, and unrepresentative data can affect model accuracy. Identifying actionable types of information, such as search and rescue requests, remains challenging.

This research combines real and generated simulated user judgements and interpretations into the classifier training process, using an active learning strategy for multi-classification problems to optimise the human-in-the-loop (HITL) process for error correction as an alternative method to address these limitations. In conjunction with a previous technique, Representation Engineering with Natural Language Explanations (ExpBERT), this technique adds feature generation from interpretation combined with original features to improve the classifier's performance [2]. The main goal of this project is to exceed the maximum achievable accuracy of the ExpBERT text classifier by employing multiple modalities of interactive systems in the classification process [3]. Highly informative unlabelled instances are queried during the iterative process using sampling strategies based on aspects corresponding to uncertainty, representativeness and diversity. Active learning is trained with a small amount of valid data, which reduces the labelling effort and the processing time in the text classification task.

During the exploration of the effectiveness of Bayesian active learning for systems, Bayesian inference for approximate neural networks, i.e. Monte Carlo dropout (MCDO), can be applied in the training of deep learning models as well as in feature extraction. The results of its stochastic forward pass (SFP) can also be used to query strategies through uncertainty evaluation. The annotator accepts and processes the extracted completed unlabelled instances for training. The valid data with labels generated through this process and the optimal hyperparameters are used to fine-tune the ExpBERT classifier, iterating iteratively to improve its accuracy eventually.

In addition, traditional neural network classification models cannot explain the uncertainty of the input samples and their distribution and, therefore, cannot give a confidence level. Therefore, this project will apply a Bayesian neural network (BNN) to reduce the uncertainty introduced by the conventional neural network in the original method through the inference method. To evaluate the performance of this method, this experiment will set up a comparison group to compare the performance of this approach with unimproved models as well as models under other algorithms, such as the modular Python library Small-Text. This includes ExpBERT classification models using different active learning strategies, models with and without Bayesian functional learning improvements, and models under other algorithms compared to a CrisisNLP dataset. The final results found that the optimised models have high accuracy and low latency.

The main conclusions of this thesis are as follows:

Lorem ipsum Lorem ipsum Blah blah blah. Lorem ipsum Lorem ipsum Blah blah blah. Lorem ipsum Lorem ipsum Blah blah blah. Lorem ipsum Lorem ipsum Blah blah blah. Lorem ipsum Lorem ipsum Blah blah blah. Lorem ipsum Lorem ipsum Blah blah blah. Lorem ipsum Lorem ipsum Blah blah blah. Lorem ipsum Lorem ipsum Blah blah blah. Lorem ipsum Lorem ipsum Blah blah blah.

Introduction

1.1 Background

Much of the focus on emergency time response systems for social platforms has been on creating better text classification algorithms to learn from the data. However, obtaining valuable annotated datasets can prove difficult [4]. The throughput of general social platform data cannot accomplish bulk annotation, so many weakly supervised forms that enable the extraction and annotation of data with high information value at a controlled cost have been widely used in classification projects [5,6]. To reduce the cost of annotation, active learning is used in the backend of social networking sites to accomplish annotation. Active learning effectively solves these problems by selecting unannotated samples with high information content to be annotated by experts [7, 8]. Querying the most informative instances using evaluation algorithms with different samples is probably the most popular approach in active learning. As a result, query strategies have naturally become a hot research topic in active learning algorithms, with various optimisation algorithms emerging. Putting annotators into an optimisation loop ultimately achieves high accuracy.

At the same time, neural text classifiers are often not comfortable with early uncertainty sampling [9]. They are often considered overconfident in their output of results, and ineffective responses to the complexity of NNs remain an essential area of research. This is because it has no inherent measure of vulnerability. In contrast, newer approaches use Bayesian extensions to achieve uncertainty estimates via drop-out or use probabilistic NNs to measure uncertainty and thus complete uncertainty sampling or model training. Also, in addition to Bayesian principles that can be applied in terms of sampling strategies [10], the difficulty of interpreting the model itself can be increased with confidence using deep Bayesian active learning.

1.2 Motivation

Against the above background, the research focuses on the accuracy, interactivity and robustness and representativeness of text classification systems.

Accuracy:

The social platforms represented by Twitter are uniformly characterised by the sheer volume of data and the amount of noise in the web language. In particular, when capturing urgent needs, it is necessary to classify categories of urgent messages and to mobilise the relevant authorities to solve the problem. However, traditional text classification systems are trained to accomplish high throughput of information. Since information texts have different distributions and the amount of information they represent can significantly affect the accuracy of the model, the results of capturing and identifying important emergency information are mostly poor [1]. Therefore, improving accuracy using semi-supervised models with active learning is essential.

Interactivity:

Traditional deep learning models lack the ability to select samples actively. In web-based text with evolutionary capabilities, traditional text classifiers with added interpretation tend to be static and therefore struggle with new, rapidly changing samples. Therefore, systems with interactivity mechanisms enable fast labelling of samples, and the performance and accuracy of the model can be further improved by interaction with the user. Particularly in emergencies, the interactivity of deep active learning can make the model more flexible and adaptable to identify and classify public safety-related information in a timely manner, providing critical support for emergency response. Thus, the importance of deep active learning in handling interactive scenarios such as the classification of urgent needs on Twitter cannot be overstated, enabling fast and accurate acquisition of tagged samples through interaction with users and providing strong support for emergency response and public safety.

Robustness:

Traditional text classification models often exhibit uncertainty and inadequate generalisation when dealing with text classification tasks. Bayesian methods can provide a statistical framework for evaluating uncertainty and using prior knowledge and a posteriori inference to enhance the interpretability of the model. Combining Bayesian methods with active learning can provide a more robust text classification system that can better deal with deficiencies in the uncertainty and generalisation ability of the model.

Representation:

The idea of active learning can help models achieve higher performance with a limited number of labelled samples. By actively selecting samples for labelling, the model can choose the most valuable samples in each iteration to improve the model's performance. In addition to this, Bayesian active learning combines the advantages of Bayesian methods, deep learning and active learning in a representative way. It can extract rich feature representations in large-scale unlabelled data, which can reduce a large proportion of bias in emergency multi-classification active learning problems.

1.3 Problem Statement

Firstly, based on the above background, we can conclude that interpretation-based neural text classification models need to be trained using a small but informative number of instances and can exploit the maximum potential of the model parameters during training. Secondly, how to select instances with significant information content to supply to human annotators and detecting the posterior performance of the instances is also the focus of this experiment. Secondly, due to the labour cost, how to simulate the processing process of human annotators is also an effective way to reduce labour. Finally, the traditional active learning query strategy is based on a fixed feature representation, where feature learning and classifier optimisation are done simultaneously in deep learning. However, fine-tuning the deep learning model in active learning alone may present inconsistencies, so exploring structural improvements in the optimisation of neural network text classifiers with ExpBERT will also have a benign impact on contingency time retrieval on social platforms by generating explanations during active learning loop.

1.4 Objects

Therefore, the following research objectives are combined with the above questions:

* To develop a pool-based human-in-the-loop active learning framework and evaluate the impact of different sampling query algorithms based on uncertainty, diversity, etc., and integrate extraction algorithms appropriately, design abort criteria and investigate whether interactive systems can improve the performance of text classification models.
* Explore the shortcomings of the optimisation architecture using Bayesian methods to improve the active learning framework and investigate whether it can effectively improve the performance of the original optimisation architecture.
* Further extension of active learning to the generation of explanations.
* Build annotator evaluation simulation algorithms and a live annotator simulation programme.
* Exploring the robustness of the advanced ExpBERT text classification architecture in a noisy environment.

1.5 Challenges

This project's main challenge is improving the training framework of the original ExpBERT textual multi-classification model to enhance performance and whether useful information can be extracted through interaction.

* The first challenge is that most active learning is applied to dichotomous problems, so it is a challenge to apply multi-classification active learning to the original framework.
* At the same time, there are multiple query strategies for active learning text classification systems, and the application and comparison of query strategies dramatically affect the performance of the active learning framework. The choice of query strategy needs to be considered in terms of time complexity and user experience.
* Several studies have been conducted on text classifiers. Still, there are few studies related to the process of interpretation generation using active learning optimization, and there is a research gap, so innovative approaches are worth being challenged.
* Finally, the assessment of the evaluation method, the development of human-in-the-loop abort criteria and the choice of hyperparameters. This significantly impacts the framework's performance on the test set and is a noteworthy aspect of this project.

1.6 Thesis Organisation

The overall thesis structure based on the research objectives consists of six modules: Introduction, Background, Design, Implementation, Evaluation and Conclusion.

1. The introduction phase in Chapter 1 provides an overview of the background of the project, presenting the motivation for the experiment in light of the background, followed by a brief description of the approach to the implementation of the overall interactive system, listing the aims of the research and the challenges faced.
2. In Chapter 2, a detailed introduction to the technology of the overall interactive system and its application will be provided, namely the application of the interactive system to a text classification system (Section 2.1), and Section 2.2 provides an introduction to the framework of active learning. Due to the multitude of example query methods in active learning, section 2.4 presents an introduction to the sampling query algorithm in combination with pseudo-code as well as formulas and the way uncertainty algorithms are evaluated and their optimisation methods in section 2.5. The final section introduces the exploratory approach of the text classification framework using Bayesian methods for improvement.
3. The overall framework and the design details of the method are described in the first section of the design chapter. Due to the importance of the query strategy, section 3.2 will focus on the query strategy adopted in this report, which has been designed to evaluate the representativeness of the examples. Finally, the design philosophy of other alternative models, such as the active learning framework incorporating the Bayesian approach, will be presented.
4. The chapter on implementation will first provide an overview of the experimental environment, in section 4.2 an introduction to the elements in the database and their properties, etc., and then in section 4.3, a detailed description of the process of implementing the simulated user operations. Since the model parameters and the sampling strategy are to be evaluated in the experiment, evaluation criteria need to be described in section 4.4. Finally, an overview of the development resources is given.
5. The chapter on evaluation begins with a description of the selection process of the hyperparameters, the presentation and analysis of the evaluation results for the validation and test sets, and finally completes with the evaluation of the effect of noise.
6. In the final chapter, a summary of the previous experiments is presented, followed by an analysis of the overall model's limitations and a discussion of further enhancements and future work.

Literature Background

2.1 Human in the loop (HITL)

The traditional natural language processing pipeline is not designed to take advantage of human feedback. However, humans are in the loop as an essential part of the interactive system, and the simulation of humans in the loop allows for the identification of model deficiencies that may not be apparent until real-world testing[11]. Godbole et al. (2004)[12] extended a text classifier based on support vector machine (SVM) active learning to naturally incorporate human input in feature engineering, term inclusion/exclusion, and term and document labelling to make statistically sound decisions. This novel interaction between human and machine learning algorithms can be called Human-in-the-loop machine learning (Munro 2020) [13]. Human in the loop can identify different HITL machine-learning solutions depending on who explicitly controls the learning process (Mosqueira-Rey et al., 2023) [14]:

**Active Learning (AL)** (Settles 2009) [15]: the system maintains control over the process of model learning, with humans acting as mediators to engage in the annotation of unlabelled data. However, humans are unable to select unlabelled data based on preferences. AL will be applied as an optimisation framework in this thesis, and the application of active learning is described in detail in section 2.2.

**Interactive Machine Learning (IML)** (Amershi et al. 2014) [16]: humans maintain a close interactive relationship with the system. For which AL and IML differ shop Dudley, and Kristensson (2018)[17] argue both AL and IML focus on selecting new points for labelling by the user, but in AL, the selection is Since IML is based on AL, they share common disadvantages. However, IML itself has the added disadvantage of mixing with Human-Computer Interaction techniques (HCI). However, IML itself has the disadvantage of being combined with Human-Computer Interaction techniques (HCI) and therefore requires unique research (Michael et al. (2020)) [18].

**Machine teaching (MT)** (Ramos et al. (2020))[19]: Training machine learning models through the guidance of human teachers. That is, delimiting the knowledge they intend to transfer to the model. Emphasis is placed on the active involvement and direction of the human teacher in the learning process. Devidze et al. (2020) [20] state that MT is more dependent on teacher expertise than active learning and less flexible regarding sample selection and handling complex tasks. Therefore, choosing the appropriate learning method according to the specific needs is crucial.

2.2 Active Learning (AL)

As the most popular learning scheme in HITL, active learning systems attempt to overcome the labelling bottleneck by asking questions to unlabelled instances and having them labelled by an expert (e.g. a human annotator) [21]. In short, active learning is the process of identifying the most informative unlabelled to hand over to an annotator, who labels and adds the already-labelled instances to the model's training process, achieving better performance by using less labelled data. In this project, the instances are large and noisy, the training task is heavy, and the accuracy is low, so active learning is applied to this project to reduce the number of irrelevant instances and increase the accuracy.

2.2.1 The AL process and scenarios

The active learning process is illustrated in Figure 1, which shows the pool-based active learning process. The machine learning model is initialised and starts by learning the labelled set of instances. L then uses the model to extract features from the unlabelled set of samples and selects unlabelled samples to provide to the human annotator according to a specific selection strategy. The labelled samples are removed from the U-set and added to the L-set to update the model. The number of labelled samples gradually increases through iterative selection, labelling and training. As the performance of the machine learning model improves, it can more accurately select the most meaningful unlabelled samples for labelling. Finally, the active learning process is terminated according to a predefined termination criterion (e.g. reaching a limit on the number of labelled samples).

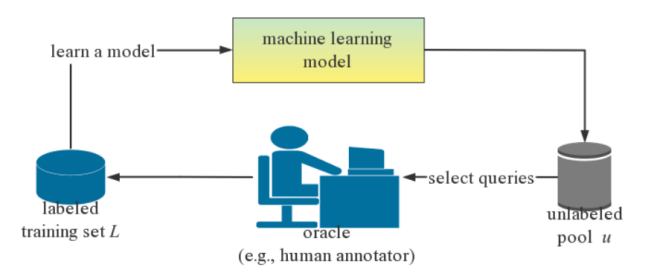


Figure 1: Pool-based active learning process

Depending on the data source, Settles (2009) [15] mentioned three main scenario settings: membership query synthesis [22, 23], stream-based selective sampling [24] and pool-based [25] AL:

**Membership query synthesis**: queries in this setting are generated by the model, and the machine learning model can request to label any unlabeled instances which are not sampled in the underlying natural distribution [15]. Efficient query synthesis is often effective for more absolute problem domains, such as tasks that employ regression prediction for absolute coordinates. This is because it can resolve the data distribution of simple d and construct reasonable data for human annotation. However, as seen in Baum and Lang (1992) [26], for complex tasks such as natural language processing, models may produce strings of text that are not easily understood, resulting in humans being unable to make judgements about these confusing texts. In the context of deep active learning, the scenario of member query synthesis can be addressed by generative adversarial networks (GANs) for data augmentation, as GANs can generate instances with a high degree of plausibility [27].

**Stream-Based Selective Sampling**: This setting is independent of the input distribution as opposed to Membership query synthesis. Stream-based sampling draws one unlabelled instance at a time from the actual distribution, and the model decides whether to request the instance's label by means of an "informativeness metric" or "query policy", corresponding to a biased random sampling [28].

**Pool-based AL**: A common setting for active learning, pool-based AL differs from stream-based sampling in that pool-based AL employs a greedy mechanism to compare the entire dataset before selecting the best query, but the latter receives the data separately for evaluation. However, when the dataset is vast, selecting the best elements for labelling may become difficult or time-consuming. Therefore, This method is suitable for the more engineering and costly manual labelling task.

2.2.2 Sampling Strategies

This section considers various strategies in active learning, and in the implementation section, the ExpBERT based text multi-classification model is used to understand the performance implications.

**Random Sampling Strategy**

Random sampling selects instances randomly, neither based on predictions nor data as well as models and is therefore used as a benchmark for the task. In this case, random sampling is used as the baseline in contrast to the more complex strategies mentioned below, especially when the pool of labels is too large [29].

**Prediction Uncertainty Sampling**

Uncertainty sampling, as the name implies, is where the active learner (model) queries the instance that is most difficult to determine its classification. In binary classification problems with probabilistic models, the posterior positive probability of such instances is closest to 0.5 [30]. For more complex multi-label classification problems, however, an entropy-based approach will be used. The more uniform the probability distribution, the higher the entropy and the higher the uncertainty of the random variables, the more informative they are. When probabilities are concentrated in a few data points, this indicates lower uncertainty and less information.



denotes the probability distribution at classification .

In the field of text classification, an alternative approach to the measurement of uncertainty is commonly used, namely, least confident [31]:



denotes the most likely class label. For binary text classification, the method is equivalent to the effectiveness of entropy-based algorithms.

In addition to the two more commonly used measures above, (Munro 2020)[13] refers to the margin of confidence, which is the difference between the two most confident predictions, and the ratio of confidence, which is the ratio of these two predictions.



 is the most confident, is the second most confident.

**Semantic-based Diversity Sampling**

Peng, Hao, et al. (2023) [32] proposed a semantic-based diversity sampling approach that can be applied to text classification. The difference with the process of measuring confidence using uncertainty sampling is that the semantic-based diversity sampling approach uses Euclidean distance to eliminate redundancy in text samples semantically. This ensures that a richer, less repetitive sample is provided to the model (learner) in the subsequent process. This abstraction approach uses the greedy k-centre algorithm of Sener and Saveravarese (2017) [33] for clustering operations. The dataset  contains  unlabelled texts and divides  into  batches, with each sample set containing  instances. It is the result of encoding the dataset. First, select vectors from  to initialise the clusters . Here examples will be considered as cluster centres. Then the k-centered algorithm searches  from , which is a set that includes members that are not in . It is the furthest from the centre of all clusters. The algorithm chosen is formulated as follows:



Where:



This is followed by updating the existing clusters to  after a loop execution and merging the output into . All text instances in  converge into a core set that best represents and generalises the dataset  in the semantic space.

**Bayesian Active Learning by Disagreement**

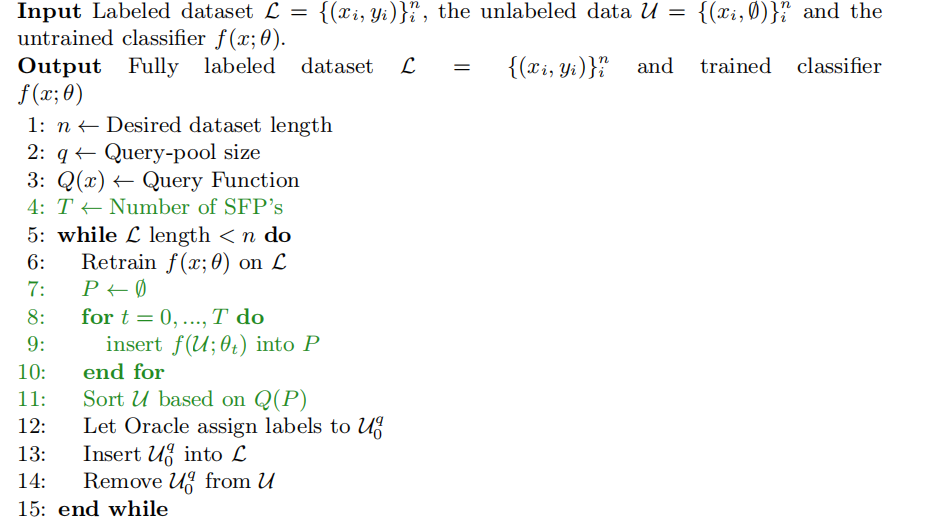
BALD (Bayesian Active Learning by Disagreement) is a method widely used in active learning for text classification. It is based on Bayesian inference and uncertainty measures and aims to select the most informative samples for annotation. In BALD, a Bayesian neural network or other Bayesian model is first used to model the text classification task. These models can estimate the probability distribution of each sample belonging to each category and provide a measure of uncertainty about each prediction. Houlsby, N (2011) [34] proposed that the critical idea of BALD is to select samples with the highest uncertainty for labelling by making multiple predictions for each sample and calculating the inconsistency between the model's predictions. The learner (model) maximises the uncertainty of the model parameters through the input .  represents the uncertainty of the target variable. The second term of the equation represents the uncertainty (entropy) for  under the condition that the parameter  obeys the posterior probability distribution  of the training data set . The average uncertainty is measured by calculating the expected value [35].



By calculating the value of BALD, samples with the highest BALD values can be selected for labelling, as these samples will provide the most significant amount of information while also minimising model uncertainty. Specifically, BALD uses Information Gain to measure the uncertainty of each sample, with a higher Information Gain indicating a higher information value of the sample. In each active learning iteration, BALD selects the samples with the highest information gain to be annotated and adds them to the annotated training set. The model is then retrained using the updated training set to improve model performance and reduce uncertainty.

2.3 AL loop with Monte Carlo Dropout

In a traditional dropout, training a neural network zeroes the output of a neuron with some probability of reducing overfitting. In MCDO, the network applies dropout [36] to enable the network to have generative nature. Multiple outputs are generated for the same input by performing multiple random forward passes. These outputs can be used in various ways to summarise the uncertainty of the model. The pseudo-code for applying the MCDO approach to the active learning loop is shown below. In MCDO, T slightly different models are created by using different Dropout samples to approximate Bayesian inference. The query function can use the results of these so-called stochastic forward propagation (SFP) to calculate uncertainty. After ranking, the instances with the highest uncertainty are assigned to annotators.



**Input** Labeled data set L, the unlabeled data U and the untrained classifier f(x).

**Output** Fully labeled data set L and trained classifier

1:n←Desired data set length

2:q←Query-pool size

3: Q(x)←Query Function

4: T←Number of SFP

5: while L length < n do

6: Retrain f(x) on L

**7: P←null**

**8: for t =0,...,T do**

**9: insert f(U) into P**

**10: end for**

**11: Sort U based on Q(P)**

12: Let Oracle assign labels to U-update

13: Insert U-update into L

14: Remove U-update from U

15: end while

Figure 2: MCDO algorithm in AL loop

2.4 Small-Text library

The text classifiers in this project tend to focus on one model and are likely to miss the application of other viable models. However, the time cost of switching models and active learning strategies, as well as the redundancy of the code, will significantly impact the progress of the experiments. Small-Text library integrates scikit-learn, transformers and PyTorch, and other common libraries that can be applied in a Python environment [37].

The architecture of pool-based active learning for text classification is shown in Figure 3, which connects the query policy, the classifier and the interface to the abort policy. Not only does it provide a state-of-the-art active learning framework for text classification work, but it also provides a range of classifiers and query policy components to facilitate active learning tasks that can be mixed and matched for rapid application in experiments and applications, making active learning easy to implement in the Python ecosystem. Small-Text offers a more flexible customisation service than the most commonly used ModAL [38] library, where the former is more focused on model integration and the selection of query strategies.

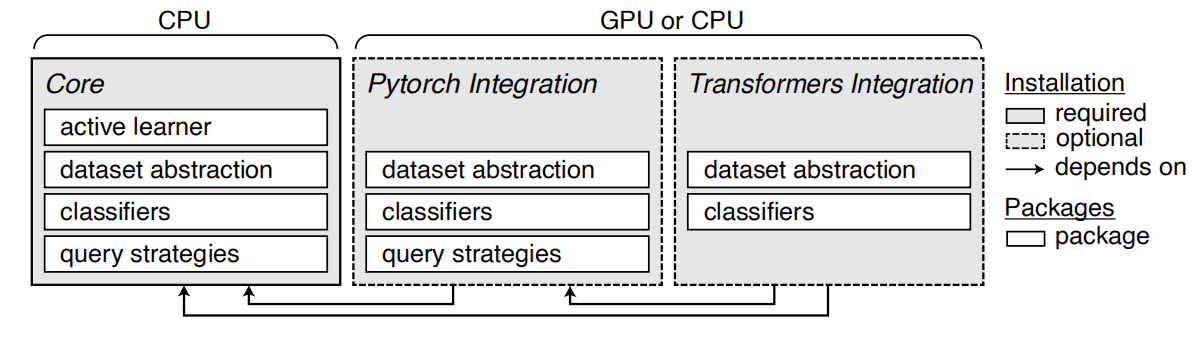


Figure 3: Framework of Small-Text

2.5 Stopping Criteria

Models are commonly trained using active learning using a query strategy similar to the greedy algorithm to select unlabelled instances. As a result, it is easy to overfit the model if the algorithm is carried through to the end, and the model does not stop training when it reaches the desired performance, wasting time and resources. Therefore, a stopping criterion in the loop is needed to prevent this.

Confidence-based Stopping:

Vlachos (2008)[39] presents calculating the classifier's confidence using the mean uncertainty on the unlabeled reference set and, for multi-class problems, using SVM classifiers with the SVM margin size as the uncertainty measure. In a loop, we look for a stopping criterion to find the maximum possible performance of the model and stop the loop [40]. However, Valchos' approach is inappropriate for text multiclassification problems, as the confidence curve stabilises after close to 500 iterations but does not show a peak due to the instability of artificial intelligence. Therefore, the peak confidence criterion based on the mean reference uncertainty is not applicable to this model.

Gradient-based Stopping:

Gradient-based Stopping addresses the drawback that Confidence-based Stopping cannot use peaks as a stopping strategy. It combines performance and uncertainty convergence stopping criteria and determines whether the active learning process should be stopped by observing the change in gradient. Precisely, the angle is calculated using the following formula:



In this case,represents the previous last n values, andrepresents the last n values. Meanwhile, a window of size k = 100 produces good results in noise mitigation while still responding fast enough to changes in the gradient [40]. Thus, the AL process is terminated when the current deterministic or estimated performance is a new maximum, whilst g is positive and falls beneath a predefined level.

Reference

1. McCreadie, R., Buntain, C., & Soboroff, I. (2019). TREC incident streams: Finding actionable information on social media.
2. Murty, S., Koh, P. W., & Liang, P. (2020). Expbert: Representation engineering with natural language explanations. arXiv preprint arXiv:2005.01932.
3. Baram, Y., Yaniv, R.E., Luz, K.: Online choice of active learning algorithms. J. Mach. Learn. Res. 5(Mar), 255–291 (2004)
4. Prabhu, S., Mohamed, M., & Misra, H. (2021). Multi-class text classification using bert-based active learning. arXiv preprint arXiv:2104.14289.
5. A. Prest, C. Schmid, and V. Ferrari, “Weakly supervised learning of interactions between humans and objects,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 3, pp. 601–614, 2012.
6. D. Tuia, M. Volpi, L. Copa, M. Kanevski, and J. Muñoz-Marí, “A survey of active learning algorithms for supervised remote sensing image classification,” IEEE Journal on Selected Topics in Signal Processing, vol. 5, no. 3, pp. 606–617, 2011.
7. B. Settles, “Active learning literature survey,” University of Wisconsin, Madison, vol. 52, pp. 55–66, 2010.
8. Y. Fu, X. Zhu, and B. Li, “A survey on instance selection for active learning,” Knowledge and Information Systems, vol. 35, no. 2, pp. 249–283, 2013.
9. Schröder, C., & Niekler, A. (2020). A survey of active learning for text classification using deep neural networks. arXiv preprint arXiv:2008.07267.
10. Ren, P., Xiao, Y., Chang, X., Huang, P. Y., Li, Z., Gupta, B. B., ... & Wang, X. (2021). A survey of deep active learning. ACM computing surveys (CSUR), 54(9), 1-40.
11. Tomaszewski, J. E. (2021). Overview of the role of artificial intelligence in pathology: the computer as a pathology digital assistant. In Artificial intelligence and deep learning in pathology (pp. 237-262). Elsevier.
12. Godbole, S., Harpale, A., Sarawagi, S., & Chakrabarti, S. (2004). Document classification through interactive supervision of document and term labels. In Knowledge Discovery in Databases: PKDD 2004: 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa, Italy, September 20-24, 2004. Proceedings 8 (pp. 185-196). Springer Berlin Heidelberg.
13. Munro R (2020) Human-in-the-loop machine learning. Manning Publications, Shelter Island.
14. Mosqueira-Rey, E., Hernández-Pereira, E., Alonso-Ríos, D. et al. Human-in-the-loop machine learning: a state of the art. Artif Intell Rev 56, 3005–3054 (2023). <https://doi.org/10.1007/s10462-022-10246-w>
15. Settles B (2009) Active learning literature survey. Tech. rep., University of Wisconsin-Madison. Department of Computer Sciences, <https://minds.wisconsin.edu/handle/1793/60660>
16. Amershi S, Cakmak M, Knox WB et  al (2014) Power to the people: the role of humans in interactive machine learning. AI Magazine 35(4):105–120. <https://doi.org/10.1609/aimag.v35i4.2513>
17. Dudley JJ, Kristensson PO (2018) A review of user interface design for interactive machine learning. ACM Trans Interact Intell Syst. <https://doi.org/10.1145/3185517>
18. Michael CJ, Acklin D, Scheuerman J (2020) On interactive machine learning and the potential of cognitive feedback. arXiv e-prints arxiv:2003.10365 [cs.HC]
19. Ramos G, Meek C, Simard P et  al (2020) Interactive machine teaching: a human-centered approach to building machine-learned models. Hum Comput Interact 35(5–6):413–451. <https://doi.org/10.1080/07370024.2020.1734931>
20. Devidze R, Mansouri F, Haug L et al (2020) Understanding the power and limitations of teaching with imperfect knowledge. In: Bessiere C (ed) Proceedings of the twenty-ninth international joint conference on artifcial intelligence, IJCAI-20. International Joint Conferences on Artifcial Intelligence Organization, 2647–2654, <https://doi.org/10.24963/ijcai.2020/367>
21. Burr Settles. Active Learning Literature Survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison. 2009.
22. Dana Angluin. 1988. Queries and Concept Learning. Machine Learning 2, 4 (1988), 319–342.
23. Ross D King, Kenneth E Whelan, Ffion M Jones, Philip G K Reiser, Christopher H Bryant, Stephen Muggleton, Douglas B Kell, and Stephen G Oliver. 2004. Functional genomic hypothesis generation and experimentation by a robot scientist. Nature 427, 6971 (2004), 247–252.
24. Ido Dagan and Sean P. Engelson. 1995. Committee-Based Sampling For Training Probabilistic Classifiers. In Machine Learning, Proceedings of the Twelfth International Conference on Machine Learning, Tahoe City, California, USA, July 9-12, 1995. Morgan Kaufmann, 150–157.
25. David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. (1994), 3–12.
26. Baum and K. Lang. Query learning can work poorly when a human oracle is used. In Proceedings of the IEEE International Joint Conference on Neural Networks, 1992.
27. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Networks, (2014).
28. Dagan and S. Engelson. Committee-based sampling for training probabilistic classifiers. In Proceedings of the International Conference on Machine Learning (ICML), pages 150–157. Morgan Kaufmann, 1995.
29. Ozan Sener and Silvio Savarese. “Active Learning for Convolutional Neural Networks: A Core-Set Approach”. In: 6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings. 2018.
30. ~~Lewis and W. Gale. A sequential algorithm for training text classifiers. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, pages 3–12. ACM/Springer, 1994.~~
31. [30] Raj, A., & Bach, F. (2022, June). Convergence of uncertainty sampling for active learning. In International Conference on Machine Learning (pp. 18310-18331). PMLR.\
32. Rong Hu, Brian Mac Namee, and Sarah Jane Delany. “Active Learning for Text Classification with Reusability”. In: Expert Systems with Applications 45.C (2016), pp. 438–449.
33. Peng, H., Guo, S., Zhao, D., Wu, Y., Han, J., Wang, Z., ... & Zhong, M. (2023). Query-efficient model extraction for text classification model in a hard label setting. Journal of King Saud University-Computer and Information Sciences, 35(4), 10-20.
34. Sener, O., Savarese, S., 2017. Active learning for convolutional neural networks: A core-set approach. arXiv preprint arXiv:1708.00489.
35. Houlsby, N., Huszár, F., Ghahramani, Z., & Lengyel, M. (2011). Bayesian active learning for classification and preference learning. arXiv preprint arXiv:1112.5745.
36. MacKay, D. (1992). Information-based objective functions for active data selection. Neural computation, 4(4):590–604.
37. Tsymbalov, E., Panov, M., Shapeev, A.: Dropout-Based Active Learning for Regression. Analysis of Images, Social Networks and Texts pp. 247–258 (2018).
38. Schröder, C., Müller, L., Niekler, A., & Potthast, M. (2021). Small-text: Active learning for text classification in Python. arXiv preprint arXiv:2107.10314.
39. Danka, T., & Horvath, P. (2018). modAL: A modular active learning framework for Python. arXiv preprint arXiv:1805.00979.
40. Vlachos, Andreas. 2008. A stopping criterion for active learning. Computer Speech and Language, 22(3):295–312.
41. Laws, F., & Schütze, H. (2008, August). Stopping criteria for active learning of named entity recognition. In Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008) (pp. 465-472).

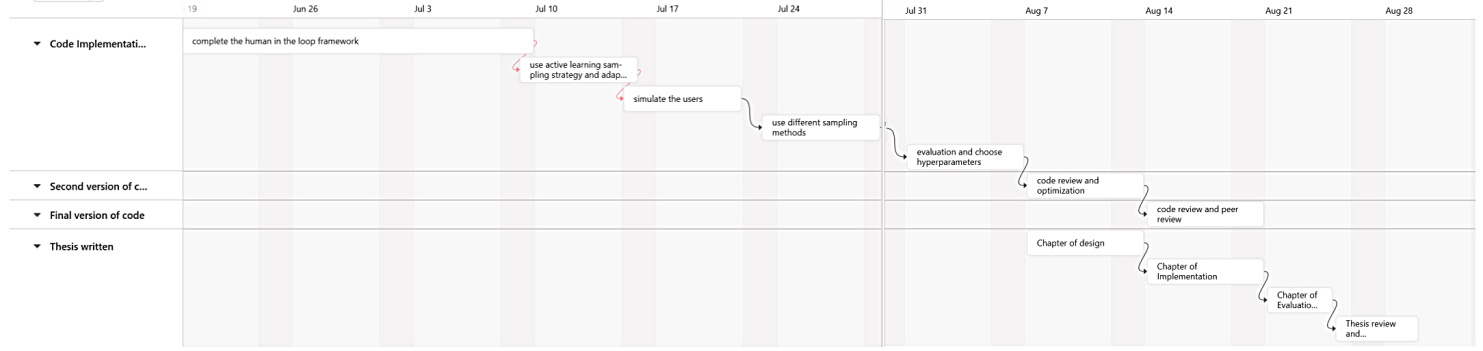
Appendix

Appendix(time plan & risk assessment)

A Project Timeline

* Week 1(Jun 26 - Jul 9): I will complete the human-in-the-loop framework use active learning
* Week 2(Jul 10 - Jul 16): I will use different sampling strategies of active learning and adapt the method to the explanation generation process with a text classifier.
* Week 3(Jul 17 - Jul 23): I will simulate the users(annotators) to give explanations to unlabeled instances.
* Week 4(Jul 24 - Jul 30): I will use more advanced deep active learning methods as a sampling strategy.
* Week 5(Jul 31 - Aug 6): I will choose the best hyper-parameters and evaluation metrics for evaluation.
* Week 6(Aug 7 - Aug 13): The first version of the code will be complete, and I will carry out the code review and optimize the algorithms; At the same time, I will start writing the Chapter on design.
* Week 7(Aug 14 - Aug 20): After two iterations of the code review, I will execute the final evaluation of the code and peer review; At the same time, the Chapter of Implementation part of my thesis will be written.
* Week 8(Aug 21 - Aug 27): The Chapter on Evaluation and Conclusion will be written.
* Week 9(Aug 28 - Aug 31): Thesis review and peer review.
* Week 10(Aug 31 - Sep 5): Oral presentation will be prepared and recorded.

The visualisation format of my task timeline is as follows:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Risk | Likelihood | Severity | Mitigation |
| 1 | Computer problem(broken or stolen) | Medium | High | Back up promptly and update the progress on GitHub. |
| 2 | GitHub server breaks down | Low | High | Back up the code on different version control platforms. |
| 3 | Body issues | Medium | Medium | Take care of body and finish the task as soon as possible. |
| 4 | Performance of model not ideal | High | High | If get problems to solve it timely and update the progress with supervisor weekly. |

Ask people for explanation (AL)

Text ?

**Next steps**

6.13 - 6.17 Finish the project plan and review the final version of my project plan

6.19 - 6.26 run the raw code and attempt the different methods (code implementation)

**Questions or things that might block the progress**

1. is there any similar literature that has implemented active learning on multi-classification problems from which the code can be reproduced

2. suggestions for project plans, ideas from other literature, and improvements

3. when PyCharm goes to implement the script, there is a default situation, prompting that the specified file cannot be found; I wonder what the problem is. Is this something that can be run directly with .sh?

4. if implemented, how long would the timeline be and how long would it normally take to complete the first version of the code?