FAAL: Feedback-Assisted Active Learning

Anonymous EMNLP submission

Abstract

Numerous increasingly sophisticated active learning methodologies have been introduced in recent years, each one with its own advantages. However, these methodologies have limited information available to them, as they rely only on signals derived from labeled data and model predictions. In this paper, we propose a novel approach that integrates user feedback signals into the active learning process, with the objective of enhancing the efficacy of existing methods. Our study demonstrates the consistent superiority of our approach, compared to traditional active learning methods when applied to diverse classification datasets and settings. Moreover, by incorporating user feedback via a contextual bandits algorithm, our proposed method exhibits significant additional improvements and robustness against user and annotator noise. We hope that these findings will encourage the adoption of strategies that incorporate user feedback in active learning, as well as broaden the inclusion of additional signals in the active learning process, thereby enabling maximization of limited human labeling resources.[1]

1 Introduction

The cost of labeling data is one of the key factors limiting the effectiveness of supervised learning (SL) in real-world applications. Active learning (AL) (Lewis and Catlett, 1994; Cohn et al., 1996; Settles, 2012) strives to address this problem by actively selecting the most informative instances from a source of unlabeled data for annotation, thereby reducing the amount of human effort needed to achieve a given level of model performance. AL methods typically leverage inherent characteristics of the data or the model (Ren et al., 2022; Liu et al., 2022; Zhan et al., 2021, 2022) to identify data points that are most likely to improve the model if labeled. However, the gains from these techniques are often marginal and inconsistent, varying significantly from dataset to dataset, and even fail to beat random sampling on some occasions (Hacohen et al., 2022).

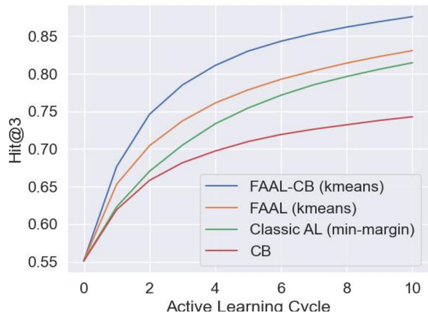

Figure 1: Average classifier hit@3 across active learning cycles over nine datasets. Our proposed methods (blue and orange) significantly outperform various classic active learning methods, including minimum-margin (green), as well as automated learning with contextual bandits (red).

While the classic AL approach relies on inherent signals to select samples for human annotation, another potential signal, often freely available in real-world applications, is user feedback, such as user clicks, star ratings, and purchase decisions. We argue that integrating user feedback signals into the active learning cycle is consistently beneficial. If users disagree with model predictions, samples can be prioritized for labeling; otherwise, they can be added to the training set. While user feedback is commonly used for automated learning techniques such as contextual bandits (CB) (Bouneffouf et al., 2020a), its deployment for active learning has not yet been explored, to the best of our knowledge.

In this paper we introduce a novel technique, denoted Feedback-Assisted Active Learning (FAAL), which leverages feedback from end-users to improve the active learning process. In contrast to CB, which solely relies on user feedback, we propose to

[1] To keep anonymity, our code will be released upon acceptance.

extend the classic AL cycle by incorporating user feedback signals in addition to human annotation. We demonstrate that adding user feedback leads to significant improvements over both classic AL and CB approaches.

We conducted experiments on multiple multi-class text classification datasets. In addition to the standard annotations, we simulated end-user feedback on the classification model predictions. As depicted in Figure 1, FAAL significantly outperforms traditional AL techniques across a range of nine datasets. FAAL is also highly effective when used together with CB, referred to as FAAL-CB, outperforming classical AL methods and automated learning with a CB algorithm by a large margin.

Our approach also exhibits high noise resilience. We evaluated FAAL and FAAL-CB under various user and annotator noise rates, finding that FAAL is robust to user noise, whereas FAAL-CB is robust to both user and annotator noise.

Thus, our contribution in this paper is threefold:

- We introduce FAAL and FAAL-CB methods that incorporate user feedback, available freely in various scenarios, as a signal to an active learning query strategy, and as an automatic augmentation of the training set through a contextual bandits algorithm.

- We show that across nine classification datasets, FAAL and FAAL-CB outperform traditional active learning approaches, showing average hit@3 improvements of 3.2% and 9.5% respectively.

- We performed an extensive analysis of the impact of noise rates on both FAAL and FAAL-CB, finding them robust and consistently superior to classic AL methods.

2 Background

In this paper, we propose that user feedback be incorporated into the active learning (AL) process to improve model learning. This strategy can also be applied in settings where the user feedback is already being leveraged to learn automatically with a contextual bandits (CB) algorithm. In Section 2.1, we provide a brief explanation about contextual bandits. In Section 2.2, we characterize user feedback. We also describe its applicability as a signal to AL and to automated learning through a CB algorithm.

2.1 Contextual Bandits

Contextual bandits (CB) are a subset of the reinforcement learning (RL) toolbox which do not keep track of the system state during an episode. Therefore, they are applicable to any decision-making problem where single-turn decisions receive a reward, rather than a full-information label. Such problems are ubiquitous in digital systems receiving user feedback, as the user often gives feedback on a small subset of the possible choices, such as products to buy and movies to watch.

As with most RL algorithms, CB algorithms aims to learn which actions lead to higher rewards. This learning process requires a balance of exploration to search for new global optima, and exploitation to maximize reward on current interactions. A simple example is the classic ϵ-greedy algorithm, which simply selects with probability $1 - \epsilon$ (exploitation) the action with the highest predicted reward, and a random action with probability ϵ (exploration).

2.2 User Feedback Overview

User feedback is found in various applications, manifesting in explicit forms like thumbs up/down buttons and star ratings, as well as implicit forms such as purchase decisions. It can be often classified into positive and negative, e.g., thumbs up and thumbs down, or, as in our experiments, whether or not a user clicks on one of the presented labels. Since it is generated by ordinary users rather than paid labelers, it is also often available in large volumes, and costs substantially less than hiring expert annotators.

Even though it may seem appealing, using user feedback may be challenging and cannot necessarily replace annotations. First, labels provide *full information*; they indicate the most accurate answer available, regardless of external factors. User feedback provides only *partial information*; it may convey only a qualitative assessment of a subset of all possible answers. Therefore, the quality of the model generating this subset may affect the feedback. Poor models, for example, may not always provide a correct choice and therefore receive mainly negative feedback, giving no indication of which of the remaining options would have been considered correct. Second, user feedback is typically noisier and less reliable than labels for various reasons. User feedback may be based on factors unrelated to model performance, such as misunder-

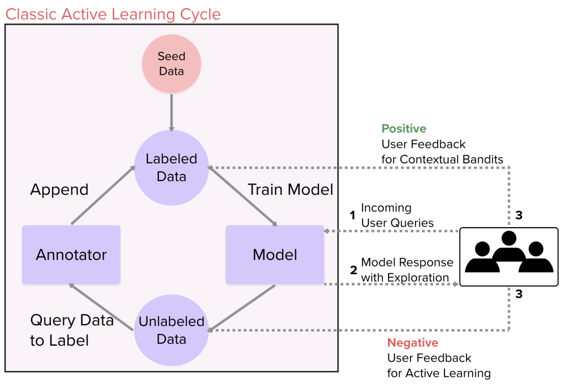

Figure 2: Our proposed outline of the extended AL cycle with FAAL/FAAL-CB. The inside of the box includes elements of the standard AL cycle. FAAL includes negative user feedback as input for the AL query strategy. FAAL-CB uses user feedback to improve the model by augmenting its training data.

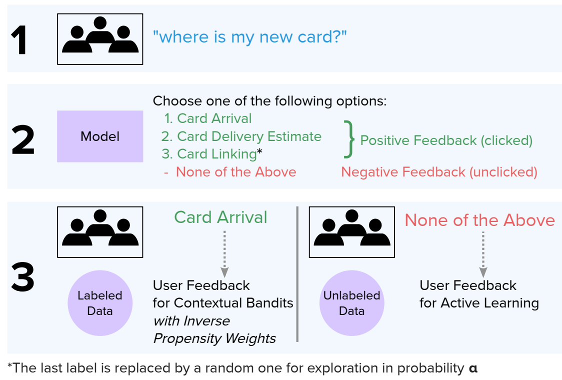

Figure 3: Illustration of a user interaction in our settings. Given a user query (blue), if the user clicks on one of the displayed labels (green), FAAL-CB utilizes this positive feedback to enhance the training set using a CB algorithm. Otherwise, if the user selects "None of the Above" (negative feedback), both FAAL and FAAL-CB prioritize the query at the annotation stage.

standing the displayed information, and dissatisfaction with the displayed content regardless of its correctness. The user may also select an incorrect label when ground truth is not displayed, and it tends to select labels closer to the top of the list (Joachims et al., 2017).

Although challenging, user feedback can enhance model training in a variety of ways. CB algorithms may use it, for example, to augment the training set with positive feedback. In addition, AL query strategies may utilize negative feedback to select data points on which the model was unable to correctly predict an answer to the user request.

3 Method

In this section we introduce Feedback-Assisted Active Learning (FAAL), a framework for utilizing user feedback as inputs for an active learning (AL) strategy. We also introduce FAAL-CB which incorporates contextual bandits (CB) algorithms as well within the AL cycle. In Section 3.1, we outline their high-level flow, and discuss how they differ from the classic AL approach. Then, in Section 3.2, we describe our implementation of FAAL and FAAL-CB given our specific multi-class classification task.

3.1 General Method

The cycle of FAAL/FAAL-CB augments the classic AL cycle with the integration of user feedback, as depicted in Figure 2. The model is initially trained on a small seed dataset of labeled examples, and then deployed to begin receiving incoming queries from users. Each query is assigned a response from the model and then receives feedback from the user. Figure 3 illustrates such interaction. These queries form a new pool from which points for labeling are selected according to an AL strategy, and then annotated and appended to the training set.

FAAL utilizes user feedback to improve the query AL strategy, for example, by prioritizing the annotation of data points that received negative feedback (see Section 2.2), so that query strategies can be directed to samples in which the model is most likely underperforming as the users disagree with model predictions for these samples. In FAAL-CB, a CB algorithm balances exploration and exploitation in the displayed options for the user, and leverages its feedback (rewards) to improve the model.

3.2 Specific Method Implementation

Given a user request, we retrieve user feedback by displaying the classifier's top-ranked k labels to the simulated user. While the number of shown labels can be dynamically adjusted, we displayed a fixed set of k labels ($k \ll$ total number of classes) for simplicity. For exploration we implemented an epsilon-greedy strategy in the last position, i.e. we replaced the last displayed label with a random one with probability α. Selecting "None of the above" is considered negative feedback ("unclicked"), while clicking one of the labels is considered positive feedback ("clicked"). This process is depicted in Figure 3.

We incorporated user feedback into the AL query strategy by first applying the AL selector on the unclicked samples from the pool. The FAAL approach is called “FAAL (random)” when unclicked samples are randomly selected, and “FAAL (kmeans)” when diverse centroids are selected for labeling using K-means.

In FAAL-CB, we apply an epsilon-greedy exploration strategy in the last label displayed to the user, and augment the training set with clicked data weighted by their inverse propensities, similarly to Joachims et al. (2017), such that clicked data gained from exploration receives a higher weight of $1/\alpha$. In order to ensure that the annotated data retains a significant influence on model training regardless of the clicked data volume, we re-weighted the annotated and clicked samples in the training set to give each 50% of the total weight.[2]

4 Experimental Setup

We evaluated our method on nine multi-class text classification datasets, which are further described in Section 4.1. For every dataset, we randomly sampled 20% of the examples and set them aside as a holdout test set. To simulate a scenario requiring minimal effort from professional annotators, we constructed the seed dataset to include only one example per class without maintaining a development set. From our evaluations, increasing the seed size to two or three per class leads to similar results.

The learning process consisted of 10 cycles, each containing 1000 randomly sampled user interactions[3]. The active learning candidates pool contained only user interactions from the last cycle to keep feedback up to date. The labelling budget per cycle was set to 50. Since we assume that users are expecting a limited number of options, we use $k = 3$. The exploration parameter was set to $\alpha = 0.01$.[4] In order to ensure the robustness of the results, we repeated each experiment using ten different random seeds.

In the absence of actual feedback from a production system, we simulate it in a manner that imitates a user behavior. For simulating ideal user feedback, we assume that when the ground-truth label is presented among the displayed k options, users would click on it (positive feedback). Conversely, if the ground-truth label is not available, users would choose “None of the above” (negative feedback). To simulate a more realistic and noisy user feedback, we introduce noise by corrupting a fraction of interactions (n) by randomly choosing a wrong option with respect to ground truth. It may be a click on an incorrect answer or on “None of the above”.[5] We set n to 20%, except for Sections 5.3 and 5.4. As opposed to the user, we assume that the expert annotator is flawless, except for Section 5.4 where we examine the effect of annotation noise, simulating it by randomly picking a wrong label instead of the correct one.

For a classifier, we used a linear SVM model taken from the “scikit-learn” library with its default hyperparameters[6] which receives as inputs the embeddings of “mini-LM-L6-v2” from the “Sentence-Transformers” library (Reimers and Gurevych, 2019). We used this classifier due to its effective performance in the absence of a validation set, which is unavailable for our specific use case. We refrained from experimenting with BERT due to the well-known instability issues associated with its fine-tuning process, as highlighted in its original introduction (Devlin et al., 2018).[7]

Dataset	# Texts	# Classes	Imbalance
CLINC150	22500	150	0.
banking77	13083	77	0.02
NLU Intent	25607	68	0.327
LEDGAR	60000	100	0.3
DBPedia l3	240942	219	0.227
Commercial1	12409	226	0.63
Commercial2	8682	168	0.84
Commercial3	22177	232	0.492
Commercial4	12755	538	0.292

Table 1: Number of texts, classes and the imbalance of the datasets. Imbalance is measured by the KL divergence between the dataset’s distribution and the uniform distribution.

[2]For more details, see Algorithms 1, 2

[3]In cases where the datasets were too small, the examples were split equally between the cycles.

[4]A higher α value would provide faster exploration and potentially a steeper learning curve, but would reduce initial system performance.

[5]We also experimented with other types of user noise, e.g., clicking the most semantically similar to the ground truth, but found the differences negligible.

[6]https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC version 1.1.1

[7]We also experimented with Tunstall et al. (2022), which is on par or better than state-of-the-art few-shot regimes on a variety of benchmarks, finding similar results (see Appendix A.3).

4.1 Datasets

We experimented with nine multi-class text classification datasets, five public and four commercial. They vary in topics, size, number of classes, and distribution across classes, as described in Table 1.

banking77 (Casanueva et al., 2020) is composed of online banking queries annotated to 77 intents. **NLU Evaluation Dataset** (Liu et al., 2019) contains 25,716 utterances simulating requests from a home robot annotated for 68 intents spanning 18 domains. **DBPedia Classes**[8] contains a taxonomic, hierarchical categories for 342,782 Wikipedia articles in 3 levels of 9, 70, and 219 classes, respectively. For our experiments, we used the last level. Our four commercial multi-class intent detection datasets, derived from chat bot classifiers, contain hundreds of intents and varying imbalance rates, thus allowing us to have a broader set of datasets. To facilitate reproducibility, we report all results also per dataset.

5 Results

In this section, we evaluate the effect of user feedback on iterative learning of a classification model. In Section 5.1, we explore the effect of adding user feedback signal to the query strategy, comparing FAAL to classic AL methods. In Section 5.2, we also leverage the user feedback to apply a contextual bandit (CB) algorithm comparing FAAL-CB to CB and AL methods. In Section 5.3, we evaluate the effect of user noise rate on user feedback-based methods. Lastly, in Section 5.4, we examine the cumulative effect of both user and annotation noise, comparing FAAL and FAAL-CB to classic AL.

5.1 User Feedback in Classic AL

First, we explore the effect of integrating the user click signal into the AL query strategy. Figure 4 shows the classifier's hit ratio across the AL cycles when randomly selecting examples to label from all the cycle candidates (random), just the clicked data (random clicked), or just the unclicked data (random unclicked/FAAL (random)). We measure hit ratio since we are primarily interested in whether the right label was displayed to the user; however, the classifier's top-1 accuracy behaves similarly (see Appendix A.4).

Labeling unclicked utterances achieved significantly better performance than labeling random utterances from the whole pool or from the clicked data across datasets (t-test $p_{value} < 0.01$). For example, in the last cycle, labeling unclicked texts improved the average performance across datasets by 5.5% over sampling from all candidates, while labeling clicked texts resulted in an average loss of 6.7%.

After establishing the effect of the user clicks signal on AL, we compared FAAL (random/kmeans) to commonly used AL methods. Table 2 shows the area under the curve (AUC) of the hit ratio across the AL cycles for each dataset. We report AUC as we are interested in the learning curve, i.e., how well the classifier performed since it was deployed and started receiving user requests. We compared FAAL to two uncertainty-based methods: least confidence and minimum-margin. We also incorporated diversity into each, similarly to FAAL (kmeans), by first clustering the pool using K-means and then selecting to label one example from each cluster using the original uncertainty-based method. Furthermore, we conducted a comparison between our method and two other popular AL methods, namely Coreset (Sener and Savarese, 2017) and BADGE[9] (Ash et al., 2020). The model's performance after training on the seed train

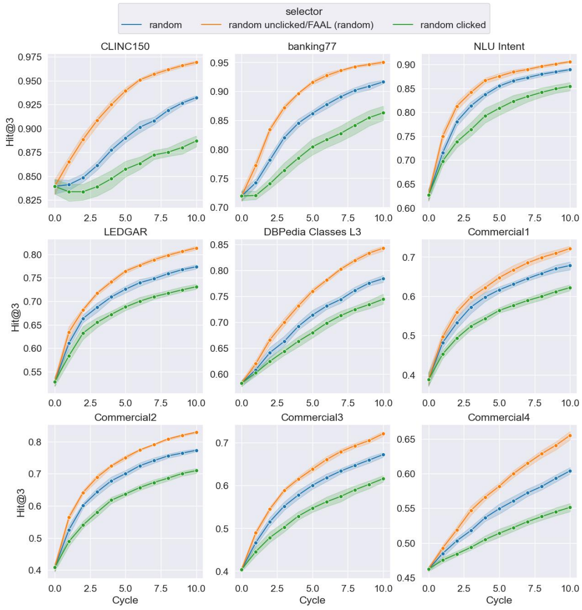

Figure 4: Incorporating user feedback in AL selection method makes a significant impact on performance. Each subplot depicts the hit ratio of the classifier when randomly sampling from all candidates (blue), the unclicked data (orange), and the clicked data (green).

[8] https://huggingface.co/datasets/DeveloperOats/DBPedia_Classes

[9] BADGE was briefly adjusted to be compatible with the SVM classifier we employ (see Appendix A.5)

	clinc	bank	NLU	ledgar	dbpedia	com1	com2	com3	com4	avg
Baseline	.839	.719	.627	.529	.583	.388	.408	.404	.462	.551
Least confidence	.922	.873	.829	.718	.735	.590	.706	.589	.563	.725
+diversity	.919	.875	.839	.723	.739	.604	.703	.595	.565	.729
Min margin	.919	.88	.844	.733	.738	.61	.71	.6	.562	.733
+diversity	.917	.877	.844	.726	.741	.611	.702	.595	.562	.731
Coreset	.895	.853	.783	.67	.681	.544	.658	.545	.546	.686
BADGE	.885	.849	.827	.71	.704	.591	.676	.582	.545	.708
FAAL (random)	.927	.888	.848	.738	.743	.624	.718	.618	.575	.742
FAAL (kmeans)	**.931**	**.901**	**.858**	**.752**	**.76**	**.643**	**.738**	**.632**	**.589**	**.756**
CB	.93	.851	.842	.685	.775	.519	.575	.547	.511	.693
FAAL-CB (random)	.962	.928	.89	.789	.845	.683	.773	.694	.603	.796
FAAL-CB (kmeans)	**.963**	**.933**	**.894**	**.795**	**.854**	**.690**	**.782**	**.703**	**.604**	**.802**

Table 2: Integrating user feedback leads to notable improvement in performance. Area under the curve (AUC) of the hit ratio across AL cycles for each dataset. "Baseline" represents performance after training on the train seed only (one example per class). Second part compares FAAL (random/kmeans) vs classic AL methods. Last three lines compare methods which incorporate CB based on user feedback. User noise level was set to 0.2.

set, is depicted as "Baseline" in Table 2.

Following Figure 4, we can see in Table 2 that FAAL (random) achieves better AUC than the classic AL methods across all datasets, achieving an average AUC of .742. Moreover, FAAL (kmeans), which incorporates user feedback as well as diversity, achieves an AUC of .756, better than all other methods except FAAL-CB (elaborated in Section 5.2). We tried to combine FAAL with other AL methods but found negligible differences. Notably, combining FAAL with uncertainty methods provided only marginal gains (see Appendix A.1). While Model prediction and user feedback are highly correlated, we believe the latter was found superior since it is extrinsic to the model.

5.2 FAAL with CB

In this section, we explore the effect of integrating user feedback by applying contextual bandits (CB), as described in Section 3.2.

We evaluated the effect of applying CB on the user feedback with and without AL in the lower part of Table 2. Most notably, FAAL-CB (random/kmeans) which incorporates the user feedback both in the AL query strategy and by applying CB, significantly outperforms all other AL methods across all datasets. While the contributions of AL (FAAL-CB vs CB) and CB (FAAL-CB vs FAAL) are clear, the difference between FAAL-CB (kmeans) and FAAL-CB (random) is more subtle, although statistically significant. Secondly, we can see that CB outperforms "Baseline" for all datasets. Therefore, with or without a labelling budget, it is beneficial to incorporate user feedback with CB. In Section 5.3, we also show that this effect is robust to higher levels of user noise.

In highly balanced datasets like CLINC150, applying only CB leads to performance that is competitive to FAAL (random/kmeans). In contrast, in imbalanced datasets such as the four commercial datasets, adding clicked data tends to further shift the distribution of the training set towards the major classes, and so AL plays a critical role. Lastly, FAAL-CB (random/kmeans) notably outperforms CB even though the number of clicked utterances (>5000) is much greater than the labeling budget (500), which differs between the two methods.

5.3 User Noise

In previous sections we evaluated with a user noise level of 20%, which was the highest level of feedback noise experimented by Gao et al. (2022a). As explained in Section 2.2, user feedback may be noisy and depends on various factors. Hence, Figure 5 depicts the effect of user noise rate on different approaches integrating user feedback. Most notably, "CB" is most influenced by the user noise rate as it relies upon it heavily in its training set. It is competitive with FAAL (kmeans) when user feedback is flawless, and comes close to "Baseline" when noise rates are high. However, in the absence of a labeling budget, CB is always beneficial.

In contrast, FAAL (kmeans), which incorporates feedback only in its query strategy, is quite robust to

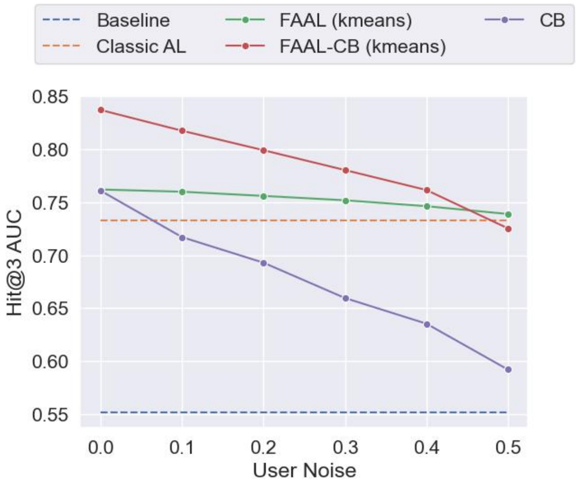

Figure 5: Hit@3 AUC of different methods as a function of user noise level averaged across datasets. "Baseline" and "Classic AL" are depicted in dashed horizontal lines as they are not affected by the user. Higher levels of user noise negatively impact feedback-based approaches, and yet FAAL (kmeans) and FAAL-CB (kmeans) surpass classic AL and "CB" outperforms "Baseline" even when user noise is high.

user noise as its performance is hardly affected. As described in Section 4, user noise flips part of the correct model responses from clicked to unclicked, and incorrect responses from unclicked to clicked. Consequently, some less useful data points, where the model already performs well, are wastefully tagged in FAAL. However, even with high noise levels, the unaffected feedback still provide a valuable signal.

Finally, FAAL-CB (kmeans), which integrates user feedback both within the AL query strategy and by applying CB, is affected by user noise, but still outperforms other approaches even when the amount of noise is over 40%. As mentioned in Section 4, when applying a CB algorithm, we moderate the user noise effect by balancing between the weight of the clicked and the annotated data.

5.4 Annotation Noise

Following Section 5.3, we experimented with different noise levels in the annotation. Figure 6 presents noise heatmaps between FAAL (kmeans) and FAAL-CB (kmeans) compared with the best performing classic AL method (minimum-margin) in Table 2. Note that the evaluated datasets may contain noise in the annotation, so the random noise we apply may aggregate over this noise.

FAAL-CB (kmeans) shows high resiliency to both user and annotation noise as it surpasses classic AL even when the noise rates are as high as 40% and 20%, respectively. We associate the robustness to noise of FAAL-CB (kmeans) to its training: As noted in Section 4, FAAL-CB (random/kmeans) provides 50% weight to both annotated and clicked data during the training process, therefore ensuring that annotated data, which is assumed to be less noisy, remains significant even when outnumbered by clicked data. As a consequence, the clicked and annotated data balance each other, enabling FAAL-CB (kmeans) to outperform classic AL, when both user and annotation noises are relatively high.

On the other hand, FAAL (kmeans), which was previously shown to be robust to user noise, is significantly affected by annotation noise, even more than minimum-margin, which is superior to FAAL (kmeans) when annotation noise is 10% or higher. We believe this result is a mirror image of the superiority of FAAL (kmeans) over minimum-margin when annotations were flawless, as shown in Table 2), i.e., FAAL (kmeans) labels more informative texts, thus their impact is greater regardless of whether they are correctly or incorrectly labeled.

6 Related Work

Active learning (AL) (Lewis and Catlett, 1994; Cohn et al., 1996; Settles, 2012) methods are usually classified as data-driven, model-driven, or hybrid (Ren et al., 2022; Liu et al., 2022; Zhan et al., 2021, 2022). Data-driven approaches utilize the structure of the data manifold to select representative or diverse samples. For example, Core-Set (Sener and Savarese, 2017) minimizes the distance to the furthest unlabeled sample. Model-driven methods rely on model predictions and uncertainty to select samples (Lewis and Catlett, 1994). Hybrid methods combine both approaches, e.g. BADGE (Ash et al., 2019) selects diverse samples holding the highest uncertainty by clustering over gradients, and CAL (Margatina et al., 2021) finds similar samples in the model feature space whose predictive likelihoods differ. Although many methods have been presented in recent years, no AL method achieves state-of-the-art results across datasets and tasks (Liu et al., 2022; Zhan et al., 2021, 2022), especially in a low-budget setting where they often perform worse than random selection (Hacohen et al., 2022).

The topic of AL with partial labels is relatively unexplored. Hu et al. (2018) focus on utilizing partial labels provided by an oracle in response

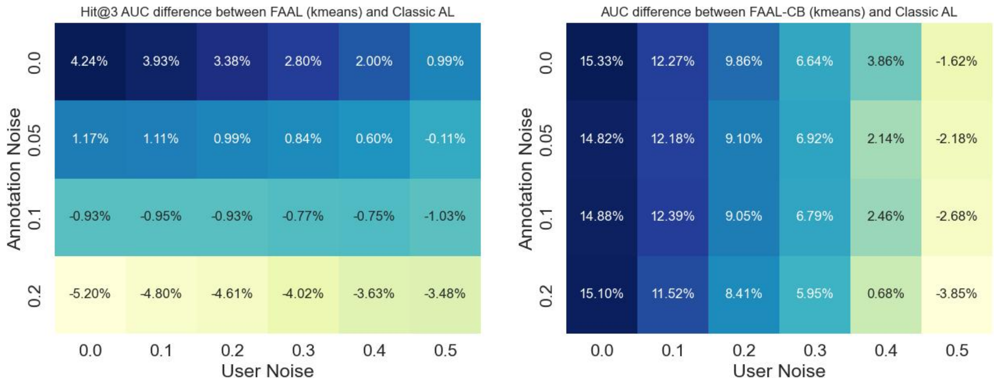

Figure 6: Hit@3 AUC difference between FAAL/FAAL-CB (kmeans) and Classic AL (minimum-margin). FAAL-CB outperforms Classic AL even in the extreme scenario of 40% and 20% user and annotation noise, respectively. On the other hand, FAAL (kmeans) is very sensitive to annotation noise, even more than classic AL.

to binary questions, assuming a hierarchical label structure. In contrast, we incorporate noisy partial labels into AL without assuming the label structure.

There is a large body of literature surrounding contextual bandits (CB) (Bietti et al., 2021; Bouneffouf et al., 2020a) and their application to learning-to-rank problems (Hu et al., 2019; Ai et al., 2018; Wang et al., 2019, 2016; Agarwal et al., 2019). One of the common approaches is Inverse-Propensity Weighting (Horvitz and Thompson, 1952; Strehl et al., 2010) with epsilon-greedy exploration (Andrew, 1999). Feedback signals from end-users have been used for automated training techniques with CB. Gao et al. (2022b) use binary feedback and a modification of the REINFORCE algorithm to help train an extractive question answering model. They also investigate the effect of noise in the user feedback signal. Suhr and Artzi (2022) explore the impact of multiple streams of user feedback of different kinds, and Zhang et al. (2019) examine combining supervised and bandit feedback data for warm-starting CB models. Other authors investigate the CB learning in a setting with missing or corrupted feedback (Bouneffouf et al., 2020b; Bouneffouf, 2021), but do not use the lack of feedback itself as a signal. Additionally, the technique of reinforcement learning from human feedback (Christiano et al., 2017; Solaiman et al., 2019) has recently exploded in popularity. Unlike our work, this technique uses the feedback as labels for a separate reward model.

7 Conclusions and Future Work

The majority of existing AL literature focuses on the advancement of intricate algorithms, aiming to maximize performance with limited data. In contrast, this paper introduces a novel approach that emphasizes the identification of *additional* informative signals to enhance active learning which yield highly effective results. Furthermore, the paper provides insight into the potential of this research area and lays the foundation for further exploration of incorporating user feedback into active learning and feedback-assisted methodologies.

Future research opportunities include exploring different ways to incorporate user feedback into other AL strategies such as BADGE (Ash et al., 2019) and CAL (Margatina et al., 2021). Additionally, other tasks may benefit from the incorporation of user feedback, for example in tasks like summarization, user feedback can be expressed in the form of star ratings in order to direct annotation efforts more effectively. FAAL can be applied to other input types, such as images and audio. One may also explore the possibility of generating pseudo user feedback with a large language model or combine LLMs with user feedback to employ FAAL in a semi-supervised setting. Moreover, a user study can also facilitate more accurate assessments of user noise and biases. Furthermore, instead of presenting a fixed number of top results to the user, future work may explore mechanisms for dynamically determining the number of options to display. This approach can optimize the user experience by providing only relevant results and minimizing unnecessary interactions. While our study utilized a unified model for AL and CB, an intriguing avenue for future research is to investigate their synchronization, as well as separating them into two different models.

Limitations

One limitation of our method is its reliance on collecting and utilizing user feedback, which may not always be possible in certain scenarios. This could be due to privacy concerns that restrict the storage of such data or the reluctance of users to provide explicit feedback, which tends to be biased towards negative responses rather than positive ones. In our proposed implementation, which relies on implicit feedback, we encounter the challenge of dealing with noise caused by biases towards the initial labels presented to the user, preferences for shown results over those not displayed, or even attraction towards intriguing labels that capture attention in addition to misunderstanding of the meaning of the label. To address this, our evaluation encompassed a broad range of user feedback. Another limitation to consider is that we only evaluated the method in one type of user feedback setting, and different types may have different effects on performance.

References

Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. 2019. A general framework for counterfactual learning-to-rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 5–14.

Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W Bruce Croft. 2018. Unbiased learning to rank with unbiased propensity estimation. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 385–394.

Alex M Andrew. 1999. Reinforcement learning: An introduction by richard s. sutton and andrew g. barto, adaptive computation and machine learning series, mit press (bradford book), cambridge, mass., 1998, xviii+ 322 pp, isbn 0-262-19398-1,(hardback,£ 31.95). *Robotica*, 17(2):229–235.

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

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2020. Deep batch active learning by diverse, uncertain gradient lower bounds.

Alberto Bietti, Alekh Agarwal, and John Langford. 2021. A contextual bandit bake-off. *The Journal of Machine Learning Research*, 22(1):5928–5976.

Djallel Bouneffouf. 2021. Corrupted contextual bandits: Online learning with corrupted context. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3145–3149. IEEE.

Djallel Bouneffouf, Irina Rish, and Charu Aggarwal. 2020a. Survey on applications of multi-armed and contextual bandits. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE.

Djallel Bouneffouf, Sohini Upadhyay, and Yasaman Khazaeni. 2020b. Contextual bandit with missing rewards. *ArXiv*, abs/2007.06368.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

David A Cohn, Zoubin Ghahramani, and Michael I Jordan. 1996. Active learning with statistical models. *JAIR*, 4:129–145.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ge Gao, Eunsol Choi, and Yoav Artzi. 2022a. Simulating bandit learning from user feedback for extractive question answering.

Ge Gao, Eunsol Choi, and Yoav Artzi. 2022b. Simulating bandit learning from user feedback for extractive question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5167–5179, Dublin, Ireland. Association for Computational Linguistics.

Guy Hacohen, Avihu Dekel, and Daphna Weinshall. 2022. Active learning on a budget: Opposite strategies suit high and low budgets. *arXiv preprint arXiv:2202.02794*.

Daniel G Horvitz and Donovan J Thompson. 1952. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685.

Peiyun Hu, Zachary C Lipton, Anima Anandkumar, and Deva Ramanan. 2018. Active learning with partial feedback. *arXiv preprint arXiv:1802.07427*.

Ziniu Hu, Yang Wang, Qu Peng, and Hang Li. 2019. Unbiased lambdamart: an unbiased pairwise learning-to-rank algorithm. In *The World Wide Web Conference*, pages 2830–2836.

Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 781–789.

David D Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *ICML*, pages 148–156.

Peng Liu, Lizhe Wang, Rajiv Ranjan, Guojin He, and Lei Zhao. 2022. A survey on active deep learning: From model driven to data driven. *ACM Computing Surveys (CSUR)*, 54(10s):1–34.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019. Benchmarking natural language understanding services for building conversational agents. *arXiv preprint arXiv:1903.05566*.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xiaolong Wang. 2022. A survey of deep active learning. *ACM Computing Surveys (CSUR)*, 54:1 – 40.

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

Burr Settles. 2012. *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

Alex Strehl, John Langford, Lihong Li, and Sham M Kakade. 2010. Learning from logged implicit exploration data. *Advances in neural information processing systems*, 23.

Alane Suhr and Yoav Artzi. 2022. Continual learning for instruction following from realtime feedback.

Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. 2022. Efficient few-shot learning without prompts.

Lequn Wang, Yiwei Bai, Arjun Bhalla, and Thorsten Joachims. 2019. Batch learning from bandit feedback through bias corrected reward imputation. In *ICML Workshop on Real-World Sequential Decision Making*.

Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 115–124.

Asaf Yehudai, Matan Vetzler, Yosi Mass, Koren Lazar, Doron Cohen, and Boaz Carmeli. 2023. Qaid: Question answering inspired few-shot intent detection.

Xueying Zhan, Huan Liu, Qing Li, and Antoni B Chan. 2021. A comparative survey: Benchmarking for pool-based active learning. In *IJCAI*, pages 4679–4686.

Xueying Zhan, Qingzhong Wang, Kuan-hao Huang, Haoyi Xiong, Dejing Dou, and Antoni B Chan. 2022. A comparative survey of deep active learning. *arXiv preprint arXiv:2203.13450*.

Chicheng Zhang, Alekh Agarwal, Hal Daumé Iii, John Langford, and Sahand Negahban. 2019. Warm-starting contextual bandits: Robustly combining supervised and bandit feedback. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7335–7344. PMLR.

A Appendix

A.1 Uncertainty vs. Click Signal

Incorporating user feedback signals into uncertainty-based methods led to a relatively minor improvement, although stable across datasets, as shown in Figure 7. The hit ratio AUC of FAAL (margin) was 0.744 while classic minimum-margin achieved 0.733. We believe this is due to the correlation between model uncertainty and user feedback, i.e., examples for which the model is uncertain are more likely to be unclicked by the user. According to our empirical analysis, 34% of the examples selected by minimum-margin were not clicked, compared to only 18% when the examples were selected by K-means.

However, we claim that negative user feedback is a stronger signal for active learning. It is not dependent on the model or train set, like signals such as a low prediction confidence. These signals suffer in the presence of model miscalibration, data that is poorly represented in the train set and outliers. Moreover, incorporating diversity in the user feedback signal improves performance substantially, which is not the case in uncertainty (see Table 2).

A.2 User Error Effect

Table 3 provides further details per dataset on the user feedback noise ratios we conducted. Clearly, the most significant relative decrease in performance was in the commercial datasets: between 32% and 35% in CB and between 33% and 18% in FAAL-CB (kmeans), comparing 0% and 50% user noise. In FAAL (kmeans), the differences were smaller, though the trends remained the same. We believe the two important factors here are the number of classes and the imbalance ratio, which affects the chances of misrepresenting a class given the noise.

A.3 SVM vs SetFit

To verify that our results can be reproduced on other classifier, we reproduced Table 2 with recently published SetFit method (Tunstall et al., 2022) that was developed for efficient few-shot learning and is on par or better than state of the art few-shot regimes on a variety of benchmarks, finding similar results, although it was not compared to recently published Yehudai et al. (2023). To train the SetFit classifier, we adopted the default settings

Algorithm 1 FAAL/FAAL-CB learning Process

Input: seed labeled data S, cycle size CS, number of cycles C, exploration chance α, labeling budget per cycle B
Output: model M

$D_{train} \leftarrow S$
$M \leftarrow Train(D_{train})$
Launch M
for $c = 1, 2, \dots, C$ **do**
    $D_c \leftarrow CollectUserFeedback()$
    $S_c \leftarrow Select(D_c, B)$
    $D_{new} \leftarrow Annotate(S_c)$
    $D_{train} \leftarrow D_{train} \cup D_{new}$
    if DoContextualBandits **then**
        $D_{click} \leftarrow \{(q, f, ips) \in D_c | f \neq N\}$
        $M \leftarrow Train(D_{train}, D_{click}, ips)$
    else
        $M \leftarrow Train(D_{train})$
    end if
end for
return M

Algorithm 2 Collect User Feedback

Input: cycle size CS, exploration chance α, model M
Output: User Feedback Data D

$D \leftarrow \emptyset$
$i = 0$
while $i < CS$ **do**
    $q \leftarrow CollectNewUserQuery()$
    $p \leftarrow M(q)$
    $s, ips \leftarrow Shown(p, \alpha)$
    $f \leftarrow UserFeedback(q, s)$
    $D \leftarrow D \cup \{(q, f, ips)\}$
    $i \leftarrow i + 1$
end while
return D

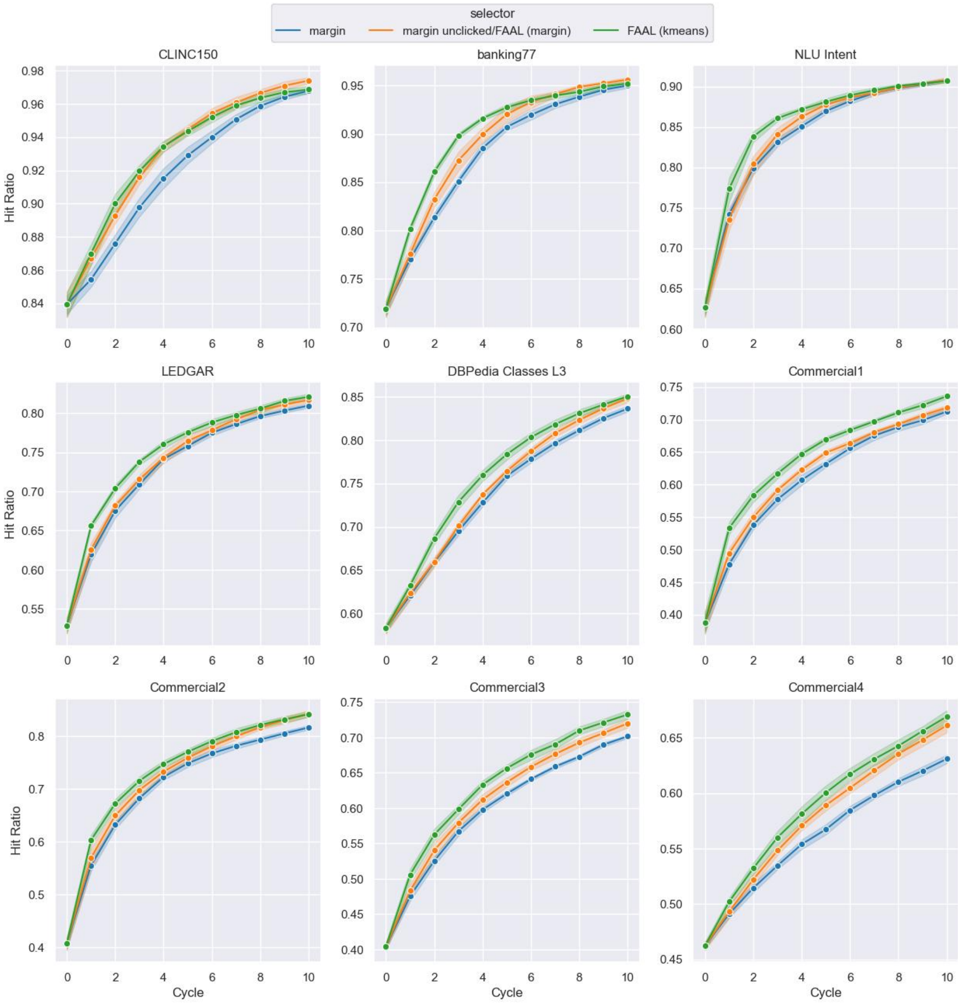

Figure 7: Uncertainty signal combined with click signal. Hit ratios across cycles for each dataset when selecting to label according to minimum-margin (orange), selecting randomly from the un-clicked data (blue), or selecting by minimum-margin from the un-clicked data (green). Adding user feedback signal on top of uncertainty-based methods leads to a minor although stable improvement in performance.

from their official git repository[10].

As you can see in Table 4, FAAL (kmeans) surpasses classic AL methods across all datasets, and FAAL-CB (kmeans) outperforms all methods significantly.

A.4 AUC of Accuracy

The main goal in our settings is to get the correct label displayed to the user, therefore we focus throughout the paper on the hit ratio metric, rather than accuracy (top-1). However, as most active learning papers reports accuracy performance, we report it for completeness.

FAAL-CB (kmeans) significantly outperforms all other methods, such as that shown in Table 2. On average, FAAL (kmeans) provides the highest accuracy, although the performance gap in accuracy is smaller than in hit ratio. We believe this is due to the fact that in FAAL we label the text that has not been clicked, thus directly maximizing the hit ratio. This is not surprising as in FAAL we label according to whether the true label was among the top predictions. However, labeling texts that the model underperformed on also improves accuracy, as FAAL (kmeans) still achieves the highest accuracy of all classic AL methods.

A.5 Implementing BADGE with Linear SVM

Since Ash et al. (2020) experimented with pseudo-gradients induced by softmax activations with respect to cross-entropy loss, and to the best of our knowledge there is no research on the efficiency of the method with other losses, we've decided to stick to their setup and code base. That is, during

[10] https://github.com/huggingface/setfit

	Noise	clinc	bank	NLU	ledgar	dbpedia	com1	com2	com3	com4	avg
FAAL	0.0	.942	.909	.855	.756	.763	.65	.746	.642	.594	.772
	0.2	.931	.901	.858	.752	.76	.643	.738	.632	.589	.766
	0.4	.922	.889	.853	.745	.749	.631	.726	.623	.579	.756
	0.5	.911	.883	.848	.737	.741	.626	.712	.62	.571	.749
CB	0.0	.958	.901	.88	.753	.837	.615	.661	.643	.6	.772
	0.2	.93	.851	.842	.685	.775	.519	.575	.547	.511	.702
	0.4	.899	.793	.789	.636	.7	.46	.488	.471	.462	.639
	0.5	.871	.757	.743	.601	.677	.406	.433	.442	.398	.595
FAAL-CB	0.0	.974	.946	.905	.818	.885	.736	.822	.756	.694	.846
	0.2	.963	.933	.894	.795	.854	.69	.782	.703	.604	.812
	0.4	.948	.91	.876	.764	.799	.647	.728	.636	.547	.773
	0.5	.927	.888	.859	.742	.774	.587	.68	.606	.466	.738

Table 3: Area under the curve (AUC) of the hit ratio for all 10 cycles for each dataset using SVM classifier. Lines 1-4 shows FAAL (kmeans) with different user noise levels with no CB. Lines 5-7 shows the AUC when applying only contextual bandits with no AL in different noise levels and lines 8-11 shows results for FAAL-CB (kmeans) with contextual bandits for different user noise levels

	clinc	bank	NLU	ledgar	dbpedia	com1	com2	com3	com4	avg
Baseline	.847	.645	.647	.569	.794	.364	.371	.36	.37	.556
Least confidence	.948	.868	.864	.808	.95	.547	.636	.579	.537	.767
+diversity	.949	.871	.872	.811	.948	.564	.649	.585	.537	.773
Min margin	.95	.868	.874	.813	.951	.564	.643	.589	.538	.773
+diversity	.947	.872	.873	.812	.948	.575	.644	.589	.541	.775
Coreset	.952	.867	.864	.805	.951	.53	.632	.575	.538	.766
BADGE	.946	.868	.875	.811	.948	.573	.649	.588	.536	.774
FAAL (random)	.95	.877	.875	.81	.95	.585	.66	.604	.551	.781
FAAL (kmeans)	**.953**	**.885**	**.88**	**.819**	**.952**	**.606**	**.669**	**.622**	**.562**	**.79**
CB	.942	.828	.819	.737	–	.456	.454	.494	.445	–
FAAL-CB (random)	.96	.891	.864	–	–	.595	.632	.627	.555	–
FAAL-CB (kmeans)	**.961**	**.895**	**.867**	–	–	**.611**	**.65**	**.636**	**.565**	–

Table 4: Table 2 but with SetFit classifier instead of SVM. Clearly, FAAL (kmeans) outperforms all other methods across datasets. Some of the runs with CB did not complete due to resources limitation.

the query process we compute pseudo-gradients as if we were using softmax after the linear layer together with the cross-entropy loss.

A.6 Complementary Results Per Dataset

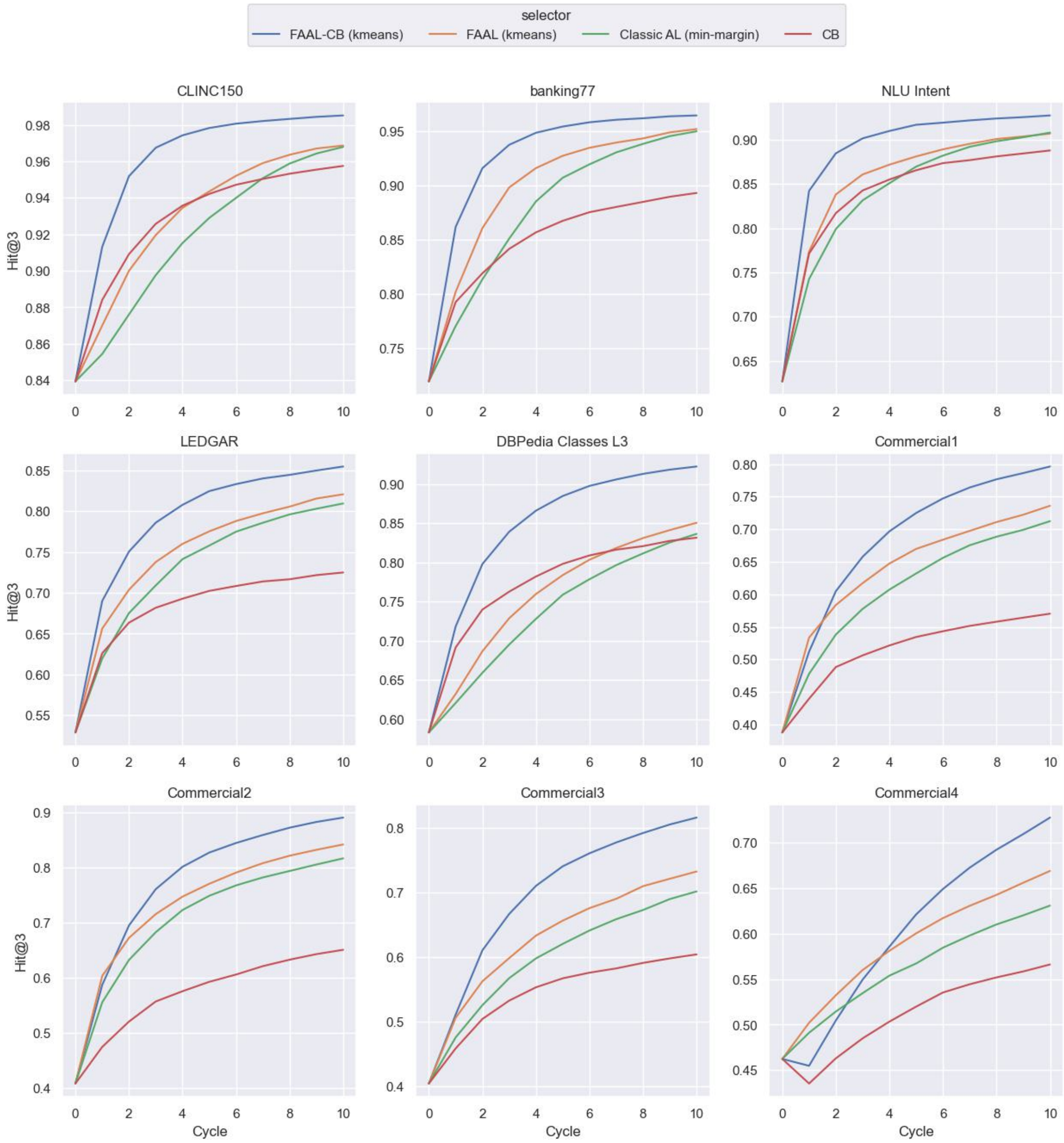

Figure 8: Figure 1 split by dataset. Notably, FAAL-CB (kmeans) and FAAL (kmeans) outperforms classic active learning approach and automated learning with contextual bandits across datasets.

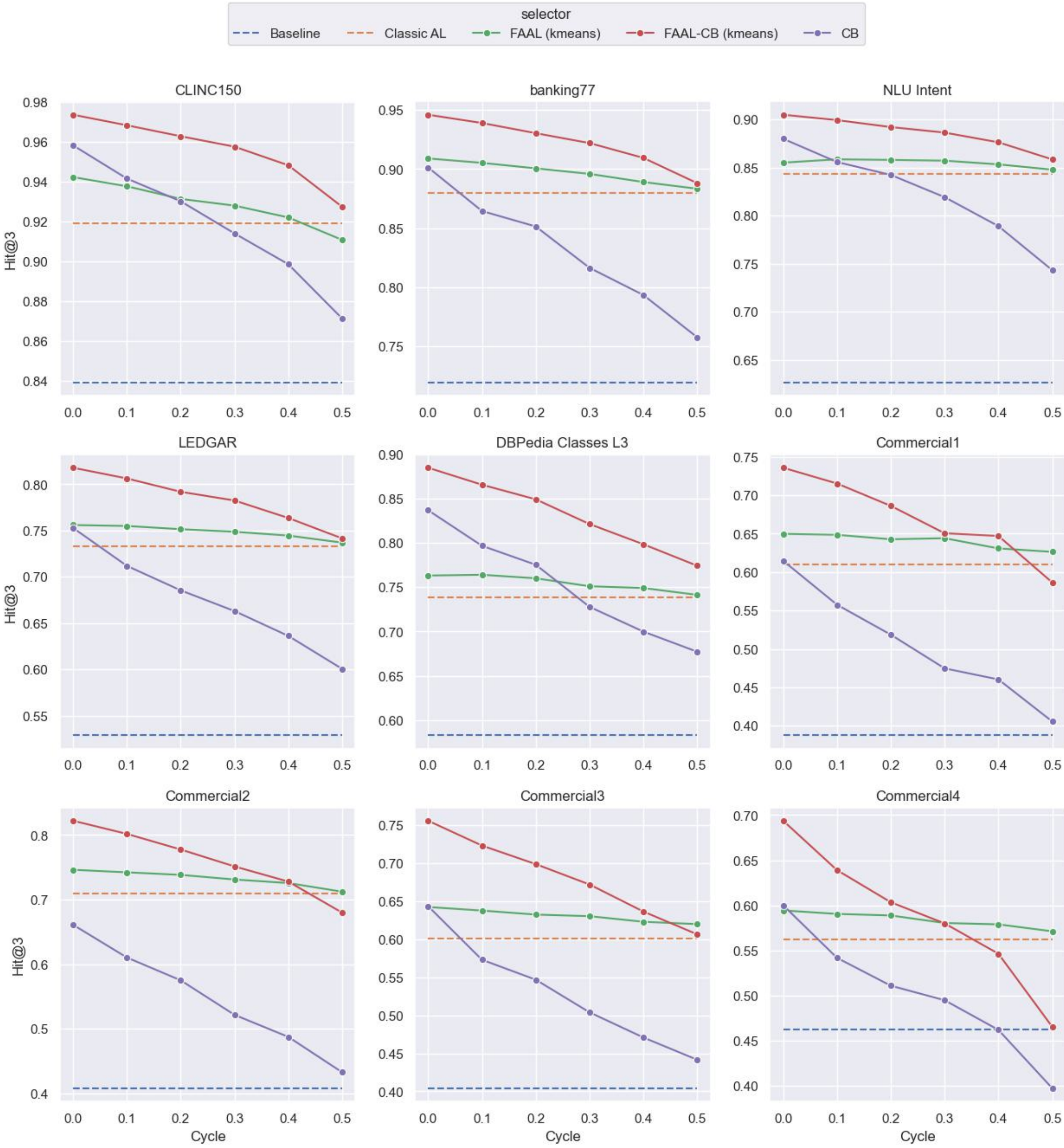

Figure 9: Figure 5 split by dataset. Similar to the original graph, FAAL-CB (kmeans) outperforms the other approaches up to a high user noise rate, although the user noise rate effect is substantially higher in the imbalanced datasets for both FAAL-CB and CB, which are highly correlated.

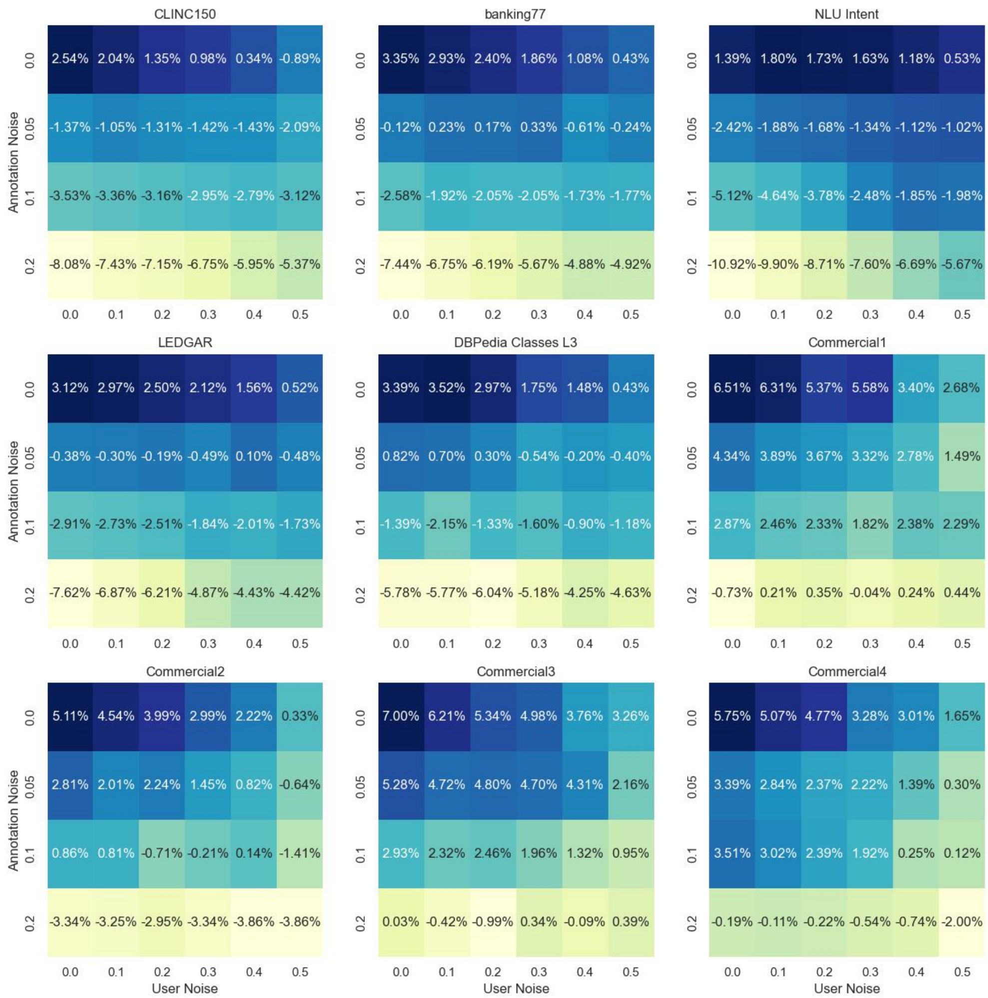

Figure 10: Left side of Figure 6 by dataset, i.e. hit@3 AUC difference between FAAL (kmeans) and Classic AL (minimum-margin) by dataset.

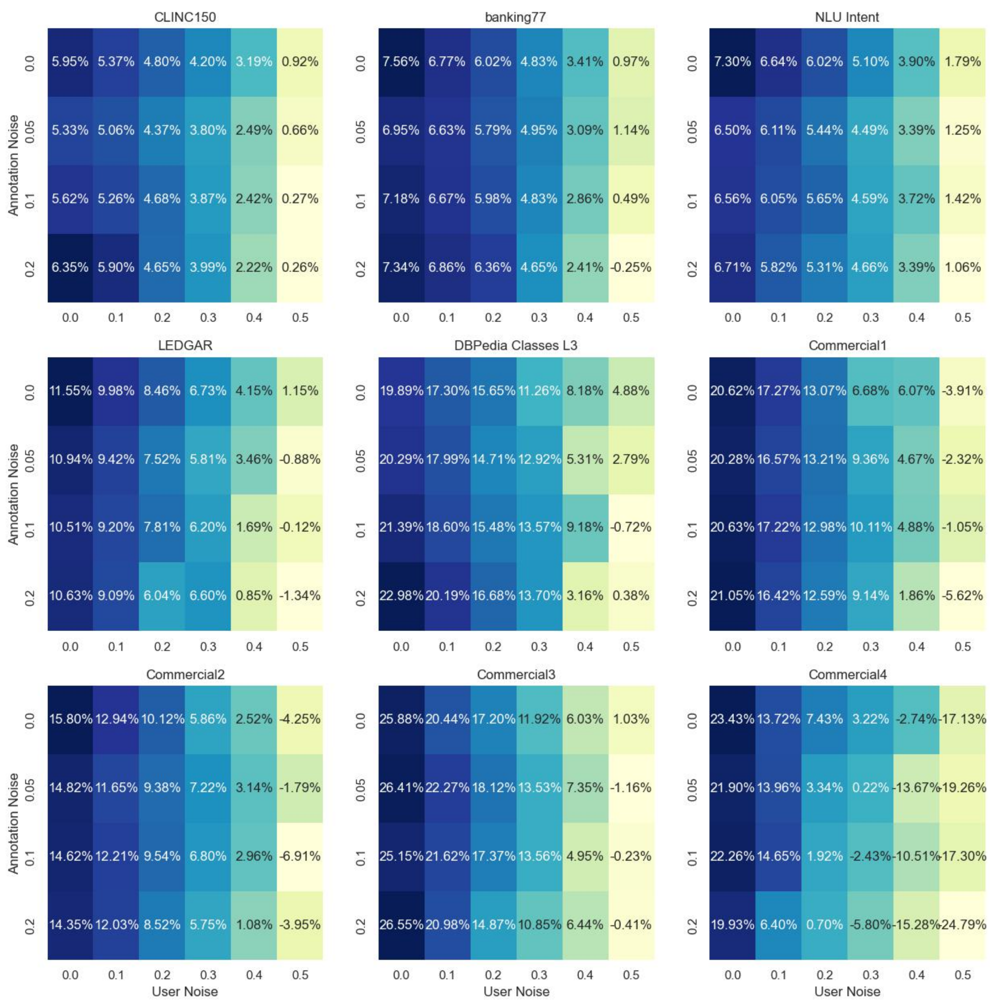

Figure 11: Right side of Figure 6 by dataset, i.e. hit@3 AUC difference between FAAL-CB (kmeans) and Classic AL (minimum-margin) by dataset.

	clinc	bank	NLU	ledgar	dbpedia	com1	com2	com3	com4	avg
Baseline	.659	.52	.418	.362	.388	.261	.294	.257	.315	.386
Least confidence	.767	.697	.624	.546	.531	.415	.512	.392	.392	.542
+diversity	.778	.716	.656	.565	.54	.431	.52	.402	.394	.556
Min margin	**.779**	.722	.67	.576	.543	.437	.527	.408	.389	.561
+diversity	.785	.724	**.677**	**.577**	**.549**	.443	.53	.404	.392	.565
Coreset	.723	.677	.555	.493	.474	.389	.493	.365	.385	.506
BADGE	.731	.687	.659	.566	.513	.427	.505	.395	.375	.54
FAAL (random)	.768	.708	.65	.557	.53	.44	.53	.411	.396	.554
FAAL (kmeans)	**.779**	**.733**	.665	.574	.545	**.456**	**.546**	**.423**	**.405**	**.57**
CB	.832	.732	.718	.563	.628	.382	.449	.386	.353	.56
FAAL-CB (random)	.877	.818	.761	.638	.687	.499	.603	.492	**.417**	.643
FAAL-CB (kmeans)	**.878**	**.821**	**.767**	**.646**	**.697**	**.501**	**.602**	**.499**	.411	**.647**

Table 5: Duplicate of Table 2 but with accuracy instead of hit@k. While FAAL-CB still outperforms other methods, the effect of incorporating user feedback in the AL acquisition function has a weaker effect on the accuracy. This is not surprising as in FAAL we label according to whether the true label was among the top predictions.