# AI Decision Systems with Feedback Loop Active Learner

Mert Kosan[1,*], Linyun He[2], Shubham Agrawal[2], Hongyi Liu[2] and Chiranjeet Chetia[2]

[1]*University of California, Santa Barbara, California, 93106, United States*
[2]*Visa Research, Austin, Texas, 78759, United States*

Abstract
Making precise decisions for high-stakes applications such as finance, health, and self-driving is critical for increasing the economy of an entity or the quality of life. In most scenarios, decision quickness is also as essential as accuracy. This is particularly true in the case of event detection problems, where late detection can cause financial or physical damage. While recent work focuses on combining fast unsupervised AI decision systems and precise human decisions to solve this problem, the quality of this cooperation remains questionable. A human can generate ground-truth labels for the AI decision systems for future improvements. However, having noisy ground truth can worsen the performance. To address this challenge, this paper proposes FLAL (Feedback Loop Active Learner), a novel bridge system between the AI decision system and human/s, designed to understand human expertise and interest using a recommender mechanism and improve AI system performance using an active learning mechanism. FLAL is able to identify human behavior and makes entity recommendations to users who can generate better ground-truth labels for these entities. Our experiments show that FLAL performs better than competing baselines and converges fast.

**Keywords**
decision systems, feedback-loop, active learning, data labeling

## 1. Introduction

Accuracy is one of the critical evaluation metrics in decision systems, especially for high-stake applications [1, 2] such as financial event detection [3], drug discovery [4], and autonomous driving [5]. Making the decision systems controlled by AI is risky because of the gray area problem [6], where AI cannot decide the actual answer and uses an artificial and pre-defined threshold. On the other hand, human decision systems are time-consuming and require an expert [7]. It leads us to the following question: Can we improve AI decision systems with the help of human expertise?

Certain high-stakes decisions, such as detecting anomalies in operating server machines wherein missing them would cause financial loss, can be easily given by AI decision systems. In

*WSDM 2023 Crowd Science Workshop on Collaboration of Humans and Learning Algorithms for Data Labeling, March 3, 2023, Singapore*
*Corresponding author.
mertkosan@ucsb.edu (M. Kosan); sherryhly@gmail.com (L. He); shuagarw@visa.com (S. Agrawal); honliu@visa.com (H. Liu); cchetia@visa.com (C. Chetia)
https://www.mertkosan.com (M. Kosan)
0000-0002-8092-5024 (M. Kosan)

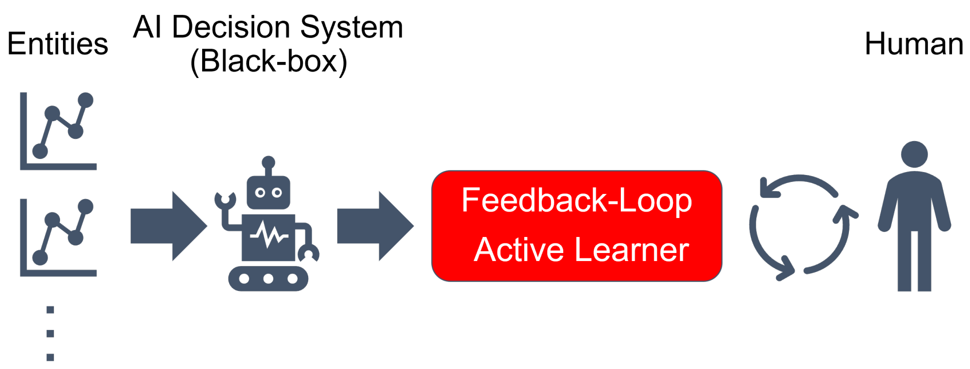

**Figure 1:** A short illustration of Feedback Loop Active Learner. It starts with multiple entities at which a black box AI system generates decisions. FLAL uses these decisions and entities to send queries to a human, who evaluates them and generates ground truth labels and feedback. FLAL uses this feedback (including human interest and expertise) in active learning training and stores the ground truth labels for future updates in the AI decision system.

order to make such decisions, AI Decision systems are created using either historical experience (previous anomaly patterns, i.e., learned features) or algorithmic design (certain behaviors are anomalies, i.e., expert-designed features). However, historical experiences are not always available because of the label scarcity problem in AI for high-stakes applications. Therefore, the anomaly decision systems generally are designed as unsupervised classification models, which affects generalizability and generates multiple misclassifications. A human decision maker may solve this problem. However, human decision making is very time-consuming and not ideal where a fast decision is necessary. For instance, if a server machine fails, AI could detect this quickly compared to a human however human expertise is needed to check and confirm the detection as well as understand its root cause. In such a context, the human required should be an expert in understanding the problem. This scenario is not limited to high-stakes decisions. Credit card approval systems or insurance acceptance systems are other examples that AI may need the help of human decisions.

Feedback loop (Human-in-the-loop) systems have been studied [8] to create a bridge between AI and humans. They collect labels from the users and improve the AI decision systems. However, can we trust the user's expertise? Even if they are experts, how do we confirm their interest in asked queries? Recommender systems have been proposed to learn the interest of people [9, 10, 11]. The system ranks unseen/unused items and recommends them to the user based on their historical interest or using user interactions [12]. While collecting ground-truth labels, the selection of humans to answer particular queries is critical to improving label correctness and quality. Combining the recommender mechanism with a feedback loop system could potentially increase the performance of AI decision systems by having plenty and correct ground-truth labels.

In this paper, we are looking at specific scenarios of multiple independent entities. Each entity has temporal multidimensional features, and the AI system makes a decision for each entity and time (e.g., anomaly/failure decision). Entities will be ranked by their relevance score to the humans and queried to them to learn their expertise and interests with a pre-defined budget. It helps the framework generate accurate ground-truth labels and labeling operation will not be challenging or boring for a human since they are interested in answering.

We propose FLAL–a novel Feedback Loop Active Learner for better ground-truth labeling–which aims to learn the expertise and interest of a human before querying the entities to them using the active learning mechanism. Figure 1 illustrates a summary of FLAL bridging between an AI decision system and a human. FLAL collects decision for entities from the AI system, ranks entities based on their relevance score to human/s, and send queries based on the budget. The human/s answers these queries and sends them back to a FLAL, which learns their behavior towards these entities as well as stores the answers as ground truths. These ground truths will be used to improve AI decision systems in the future. Our main contributions can be summarized as follows:

- We highlight the limitation of current AI decision systems, human decisions, and their cooperation to generate data labels. AI decision system makes many mistakes, human decisions are slow, and cooperation may be limited because of the lack of expertise or interest from humans.
- We propose FLAL, a novel feedback loop active learning framework, for better ground-truth generation and understanding of human behavior. It uses active learning to train the framework based on human feedback and stores generated data labels to improve AI decision systems in the future.
- We conduct experiments to verify the effectiveness of FLAL. We show that our framework performs better than competing baselines: random forest active learner, AI decision-based, and random recommendations. FLAL not only has the best performance but also converges fast.

## 2. Related Works

**Human-in-the-Loop**

Human-in-the-loop, in other words, feedback-loop, mechanisms are studied in the literature to enhance AI performance by label annotation [13, 14] and generating explanations [15, 16, 17, 18] to black-box operations. Since feedback-loop systems are generally real-time systems, they often use active learning during their training [19, 20, 13]. In our work, we also adapt similar ideas by incorporating human decisions into AI. However, our method increases the efficiency of this cooperation by learning the expertise and interest of humans before asking them questions.

**Recommenders**

Recommendation systems are one of the solutions for understanding the behavior of individuals. They are designed to infer interests and recommend items to humans based on their historical experience or user interactions [12]. The main idea is to rank all relevant items to the user and recommend the top ones. Therefore, ranking algorithms become one of the main components [21, 22, 23] in designing recommendation systems. Recently, as opposed to classical recommenders such as collaborative filtering and matrix factorization, deep-learning frameworks are applied to learn a better representation of items [10, 24, 25]. However, a lack of data availability can restrict the number of parameters to be learned. Even though we still use the advantage of deep learning recommenders, we keep our framework simple but effective. Our recommender system finds better queries based on the user's expertise and interests. In this way, the feedback loop will generate better ground truth labels.

**Temporal Embeddings**

Representation learning on time-series data has become a popular technique to reduce the dimension of the temporal data while keeping the representation (meaning) of it intact [26, 27]. Time2Vec [28] uses a sine activation function to embed the time-series data. Unsupervised time-series embedders have been proposed [29, 30] to deal with label scarcity. Franceschi et al. [29] use the triplet loss function to learn a representation of multidimensional time-series data. The trained embedder can embed any time-series data with any length. More recently, Zerveas et al. [30] propose a transformer-based framework by reconstructing the mask part of the time-series data. FLAL uses a pre-trained temporal data embedder to represent time-series data coming from the entities for better and more compact representations.

## 3. Methodology

### 3.1. Problem Formulation

We formulate our problem as a self-supervised time-series classification. Given an entity set $\mathcal{E} = \{E_1, E_2, \ldots, E_n\}$ where each entity represents a multivariate time-series data (i.e., $E_i = [x_{i1}, x_{i2}, \ldots, x_{it}]$), an unsupervised AI decision system $\mathcal{D}$ which generates decision probability $d_{it}$ for each entity and timestamp (i.e, $\mathcal{D}(E_{it}) = d_{it}$), a user set $\mathcal{U} = \{U_1, U_2, \ldots, U_m\}$, and interest labels $\mathcal{Y} \in \{0,1\}^{m \times n \times t}$ for each user to certain timestamp and entity; our goal is to learn a function $\hat{F} : \mathcal{E}, \mathcal{D} \to \mathcal{Y}$ that approximates the expertise of users.

### 3.2. FLAL: Feedback Loop Active Learner

We introduce FLAL, a feedback loop active learner that generates ground-truth labels for the AI decision system while learning human expertise and/or interest. FLAL performs user mapping and feature extraction that optimizes the human expert's predictions. As a result, it obtains better ground-truth labels for the AI decision system.

Figure 2 describes the steps of FLAL in detail. FLAL finds a global embedding space using pre-trained time-series embedders and translates the global embedding space into personalized embedding space. To overcome the cold-start problem, FLAL extracts features from user embedding space and incorporates AI decisions into this feature set. Finally, it calculates relevance scores for each entity and sends the top $Q$ to human experts for evaluation. The human classifies each query which in turn generates ground truth information and adds feedback (explicit or implicit) about their expertise or interest in a certain query. FLAL trains its framework using this feedback via an active learning mechanism and stores the ground truth information to improve AI decision systems in the future when necessary.

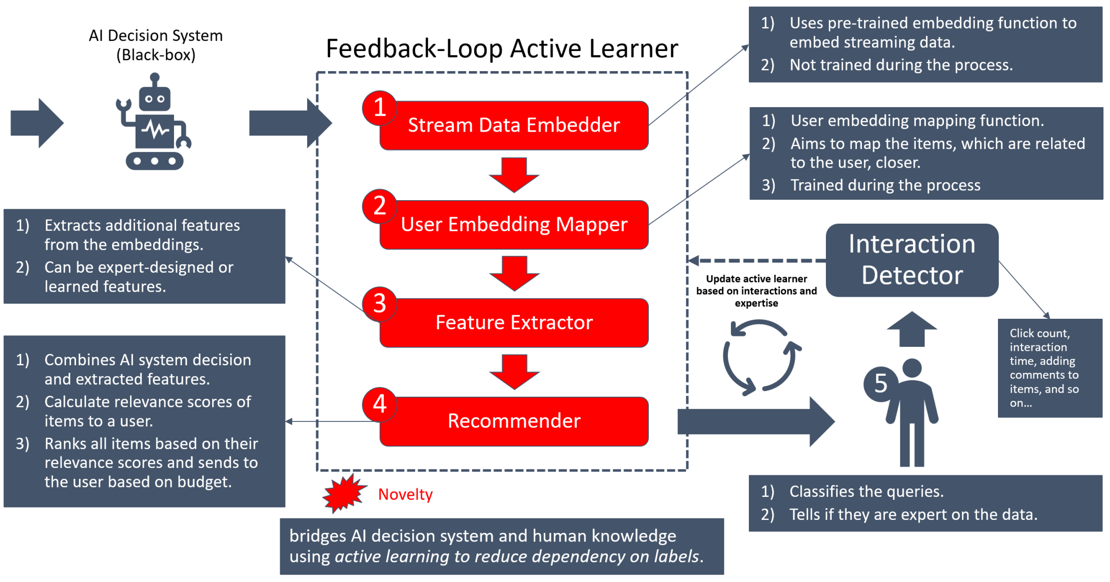
AI Decision System (Black-box)
Feedback-Loop Active Learner
1 Stream Data Embedder
2 User Embedding Mapper
3 Feature Extractor
4 Recommender
1) Uses pre-trained embedding function to embed streaming data.
2) Not trained during the process.
1) User embedding mapping function.
2) Aims to map the items, which are related to the user, closer.
3) Trained during the process
1) Extracts additional features from the embeddings.
2) Can be expert-designed or learned features.
1) Combines AI system decision and extracted features.
2) Calculate relevance scores of items to a user.
3) Ranks all items based on their relevance scores and sends to the user based on budget.
Interaction Detector
Update active learner based on interactions and expertise
Click count, interaction time, adding comments to items, and so on...
5
1) Classifies the queries.
2) Tells if they are expert on the data.
Novelty
bridges AI decision system and human knowledge using *active learning to reduce dependency on labels*.

**Figure 2:** Feedback Loop Active Learner steps. (1) It starts with embedding stream data using pre-trained embedders. (2) User embedding mapper maps the embedding space into a more personalized space. (3) Feature extractor generates learned or expert-designed features to tackle the cold-start problem for recommenders. (4) Generates relevance scores based on AI decision system and extracted features. It sends queries to users for ground truth generation. (5) User generates ground truths and relevancy of the query. They send them back to the framework. Later, FLAL updates its components using an active learning mechanism and keeps ground truth information for future updates on the AI decision system. Notice that the user's interest (relevancy) in queries can also be inferred using interaction detectors.

### 3.2.1. Stream Data Embedder

To increase the expressiveness and compactness of our data, we use an unsupervised multivariate time-series embedder to represent the entity's time series. This part of our algorithm is pre-trained with the data, which is not used in our experiments. We used [29] for our stream data embedder since it is more flexible to different time-series lengths and generates good representations for anomaly data compared to [30]. During the embedding of the time-series data, we consider the last $\tau$ timestamps.

$$h_{it} = S([x_{i(t-\tau)}; \ldots ; x_{it}]) \tag{1}$$

where $h_{it}$ is an embedding of $E_{it}$, $S$ is an unsupervised multivariate time-series embedder, $x_{it}$ is multivariate time-series data for $E_{it}$, and $[\cdot;\cdot]$ is the row concatenation operator.

### 3.2.2. User Embedding Mapper

Global embedding space may not be as representative as user embedding space, where one can understand the expertise of users. Therefore, we have a user embedding mapper that maps generated embeddings from the stream data embedder to the user embedding space. This allows them to distinguish between relevant and irrelevant entities to the user. Figure 4 shows

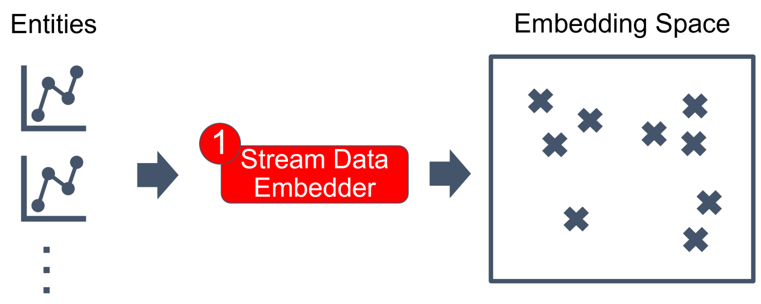

**Figure 3:** An illustration of Stream Data Embedder. The pre-trained unsupervised embedder takes input from all entities' time series data (up to $\tau$ timestamp history) and embeds them into $d$ dimensional space. This allows a better and more compact representation of time-series data.

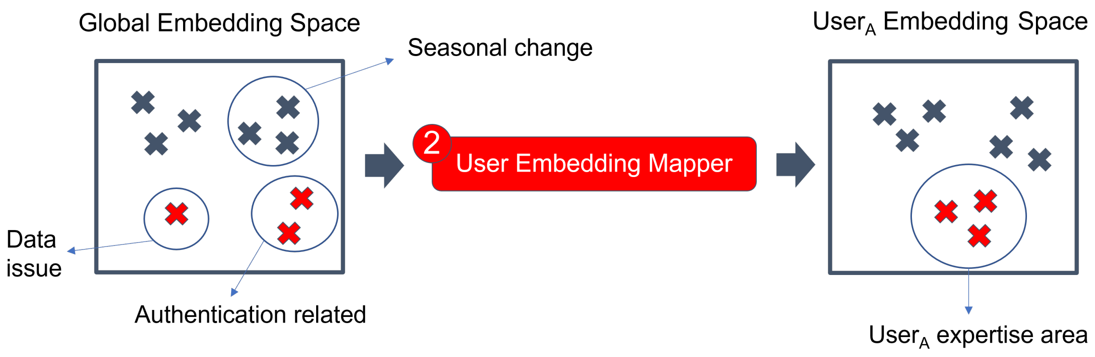

**Figure 4:** The usefulness example of user embedding mapper. Since the global embedding space will group related entities together and a specific user may be interested in different types of entities, the user embedding mapper will learn how to regroup entities. This example shows that the algorithm may detect the anomalies for different reasons: data issues, seasonal change, and authentication-related. Therefore, if only data issues and authentication-related anomalies (red entities) are relevant to the user, the algorithm groups them together to make the embedding space personalized.

an example of the usefulness of user embedding mapper. For an anomaly detection problem, multiple reasons can cause an anomaly. But the users are often experts on a certain subset of those anomalies and the user embedding mapper will map these types close to each other. As a result, they can be separated from the other anomaly types or normal ones. The user embeddings are generated as follows:

$$h_{it}^{A} = g_A(h_{it}) \tag{2}$$

where $h_{it}^{A}$ is a user embedding for User A, and $g$ is the user embedding mapper. $g$ can be designed as any function, such as the identity or neural networks.

### 3.2.3. Feature Extractor

Since we do not know any information about the users and cannot conduct an initial survey as most recommenders do, we need to extract features from the user embedding space. Features can be learned or designed for a specific application scenario. An example of an expert-designed feature for anomaly applications can be the average distance from one item to others, which will likely be higher for anomaly cases. However, learned features are shown as more expressive than expert-designed features because it is hard to design or engineer all useful features. Feature extractor can also be seen as a function layer on top of the user embedding mapper. So it will learn new features from $h^A_{it}$:

$$h'^A_{it} = f(h^A_{it}) \tag{3}$$

where $h'$ has smaller dimension than $h$, and $f$ is a feature extractor function. $f$ can also represent a set of learned and expert-designed functions. In that case, $h'$ will be a concatenation of extracted features.

### 3.2.4. Recommender

In order to find better queries for specific users, we calculate each entity's relevance score to a user. Note that the AI decision system $D$ already calculates decision probability $d_{it}$ for entity $i$. Even though this probability may not be fully correct, we can incorporate it into relevance score calculation to ease the cold-start problem. Furthermore, we will also use the extracted features from the feature extractor. The relevance score of $E_{it}$ for a user $A$ calculation will be as the following:

$$r^A_{it} = w_1 \times d_{it} + \sum W \odot h'^A_{it} \tag{4}$$

where $w_1$ and $W$ are learned weights. Also, these weights may tell us a story about the importance of AI decision systems and extracted features for different users. Once relevance scores are calculated for all entities, the recommender will send the top $Q$ relevant entities to the user to get feedback.

### 3.2.5. User Feedback

The users will have a list of queries to be checked and answered. The user will respond to each query with two answers: (1) what should be the decision of the AI system? (2) what is their expertise/interest in this query? The first answer is stored to update the AI decision system if necessary, while the second is used to train FLAL. Note that our main algorithm is not controlling the answering part done by the user. If a query has no response, it means no decision (i.e., do not use to improve the AI decision system) and no expertise (improve FLAL with the information that the user is not an expert). Furthermore, an interaction system can be designed to understand the expertise or interest of the user in queries by looking at their click count or other related metrics. However, this is out of the scope of this project. The expertise information will be stored in $e^A_{it} \in \{0, 1\}$, and used to update FLAL.

## 3.3. Training FLAL

We train our framework based on active learning principles since the problem requires learning the behavior of users in real-time to collect better ground truth labels for AI decision systems. The collected ground truth information will be stored to update the AI decision system if necessary. Our active learning mechanism focuses on updating the recommender using feedback from expertise information. For each timestamp t, we train our objective which aims to sort relevance scores of entities, $R_t^A = [r_{1t}^A, \dots, r_{nt}^A]$ based on the expertise information array $E_t^A = [e_{1t}^A, \dots, e_{Qt}^A]$. More specifically we use a contrastive loss for our active learning training which contains three different terms as follows:

$$\begin{aligned}\max L_{\text{ALL}}(E_t^A, R_t^A) = \; & x_1 * \sum_{\substack{i<Q \\ e_{it}^A=1}} \sum_{\substack{Q>j>i \\ e_{jt}^A=0}} \sigma(r_{it}^A - r_{jt}^A) \;\; \rightarrow L_{\text{WIDEN}} \\ + \; & x_2 * \sum_{\substack{i<Q \\ e_{it}^A=1}} \sum_{\substack{Q>j>i \\ e_{jt}^A=0}} \sigma(r_{it}^A - r_{jt}^A) \;\; \rightarrow L_{\text{NARROW}} \\ + \; & x_3 * \sum_{\substack{j<Q \\ e_{it}^A=1}} \sum_{k>=Q} \sigma(r_{kt}^A - r_{jt}^A) \;\; \rightarrow L_{\text{RECOVER}}\end{aligned}$$

where $L_{\text{WIDEN}}$ widens the gap between correctly ranked positive and negative samples, $L_{\text{NARROW}}$ narrows the gap between wrongly ranked positive and negative samples, and $L_{\text{RECOVERY}}$ recovers unrecommended by narrowing the gap between wrongly-recommended samples and unrecommended samples (which may contain useful recommendations). We use $x_{1,2,3}$ as a tunable hyperparameter to value each term respectively. They can be optimized based on the scenario and the need of an application.

### 3.3.1. Example scenario for our training

Let $R_t^A = [1.5, 1.2, 1.1, 0.9, 0.3, 0.2]$, $Q = 4$, $E_t^A = [1, 0, 0, 1, ?, ?]$, $x_1 = 0.50$, $x_2 = 0.75$, and $x_3 = 0.25$, then our objective will be calculated as follows:

$$\begin{aligned}L_{\text{ALL}}(E_t^A, R_t^A) &= 0.50 * (\sigma(1.5 - 1.2) + \sigma(1.5 - 1.1)) \rightarrow L_{\text{WIDEN}} \\ &+ 0.75 * (\sigma(0.9 - 1.2) + \sigma(0.9 - 1.1)) \rightarrow L_{\text{NARROW}} \\ &+ 0.25 * (\sigma(0.3 - 1.2) + \sigma(0.2 - 1.2) + \\ &\qquad \sigma(0.3 - 1.1) + \sigma(0.2 - 1.1)) \rightarrow L_{\text{RECOVER}}\end{aligned}$$

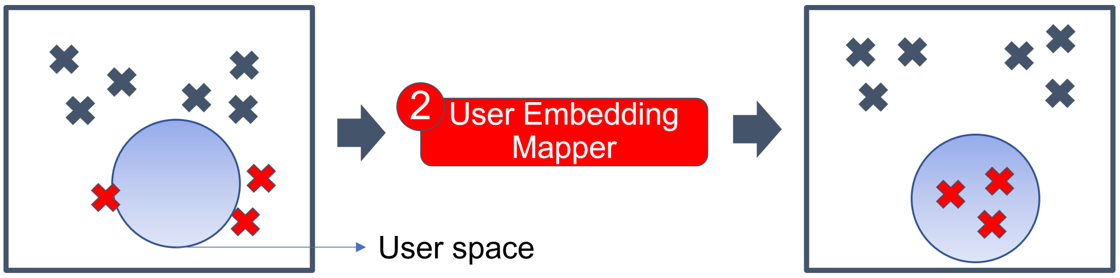

**Figure 5:** The example scenario of the user simulation. The user space will be randomly assigned in the embedding space. The entities inside of this space will be relevant to the user. So user embedding mapper should learn how to map relevant items to this space based on the answers by the user. This will allow better ground truth generation by the user.

### 3.4. User Simulation

To see the effectiveness of our feedback-loop part, we need to simulate the user answers. One way to do this is by using ground truth information for the AI decision system if it is available (simulating that the user's expertise/interest is the ground truth). We apply this strategy in this paper. However, this would allow only one user available in the system. To extend the number of users to more than one, we propose a new way of simulating users. Each user is represented as a Gaussian latent space in the entity embedding space. The user space is assigned randomly. If a query entity is in this space, the user is considered an expert. The user embedding mapper will map related entities into this space. Figure 5 shows the mapping example.

## 4. Experiments

We illustrate the empirical verification of FLAL compared to three baselines on a dataset. First, we compare method performance with precision at Section 4.3. Later on, we conduct two ablation studies: the effect of user embedding mapper (Section 4.4) and objective function weights (Section 4.5).

### 4.1. Dataset

We use a public Server Machine [31] dataset for our experiments. The dataset contains 38 time-series data with various lengths. For our purpose, we chucked the data into 100 time series (entities) with a length of 365. Each time series consists of 38 different features. At any point in the time series, machine activity is classified as normal or failure. Failure represents an anomaly.

### 4.2. Experimental Settings

#### 4.2.1. Baselines

We used three baselines to compare our method.

**Random:** It makes random recommendations in the recommender step to the user.

**AI System Decisions:** It only uses AI decision system probability to recommend entities to the user.

**Random Forest Active Learner:** It combines uncertainty and confidence scores for each entity and recommends them to the user. The model is trained using active learning with a random forest as the estimator and the same settings as FLAL.

#### 4.2.2. Other Settings

**Model selection:** We select user embedding mapper as a linear layer as a result of ablation study (See Section 4.4), feature extractor generates $15$ learned features with linear layer, and recommender is also a linear layer to generate a relevance score for an entity. To simulate user feedback, we use the ground truth information of the dataset.
**Hyperparameters:** We tune hyperparameters of FLAL using grid search. We optimize our model using Adam optimizer with a learning rate of $0.0001$, L2 normalization weight on model weights $0.001$. We select loss function weights $x_1$, $x_2$, and $x_3$ from $\{0.0, 0.5, 1.0\}$ (See Section 4.5). In our experiments, $\tau$ is set to 127 (the length of the multivariate time series data becomes 128 with the current snapshot) and the embedding size $d$ is 128. We set $Q$ to 10 and 20 for different runs. The number of recommended items is also set to Q.
**Evaluation metrics:** Since the real evaluation can only consider feedback from the recommended entities, we compare precision metrics in our experiments. At each round, we calculate precision@Q and average precision@Q.

### 4.3. Performance

Figure 6 shows precision@Q and average precision@Q, where Q is 10 and 20. We calculate a cumulative average of precision performance at each step. Note that this performance will reflect improved AI decision system performance since we use the user's interest/expertise label as a ground truth decision. Our method, Feedback Loop Active Learner, outperforms the competing baselines at all reported metrics, especially after 10-20 steps. Another essential requirement of learning human interest is convergence speed. FLAL converges in around 50 steps, faster than the best baseline Random Forest Active Learner.

Notice that the precision performance of the AI Decision System is not enough. However, active learner mechanisms improve the performance of the decision system drastically, while random decisions are still worse than the original AI decision system. Another notable difference between $Q = 10$ and $Q = 20$ is in precision@Q performance. When $Q = 20$, the performance drops below 0.6. However, this essentially could happen because the number of anomalies in the data is hardly more than $12$ (i.e., $20 \times 0.6$) at a certain time. This shows us that we should optimize the number of recommended entities based on the number of anomalies at every step instead of using the same values as the budget of $Q$. The average precision is less vulnerable to this issue since the leading zeros do not affect the result. For both $Q$, the performance is close to each other.

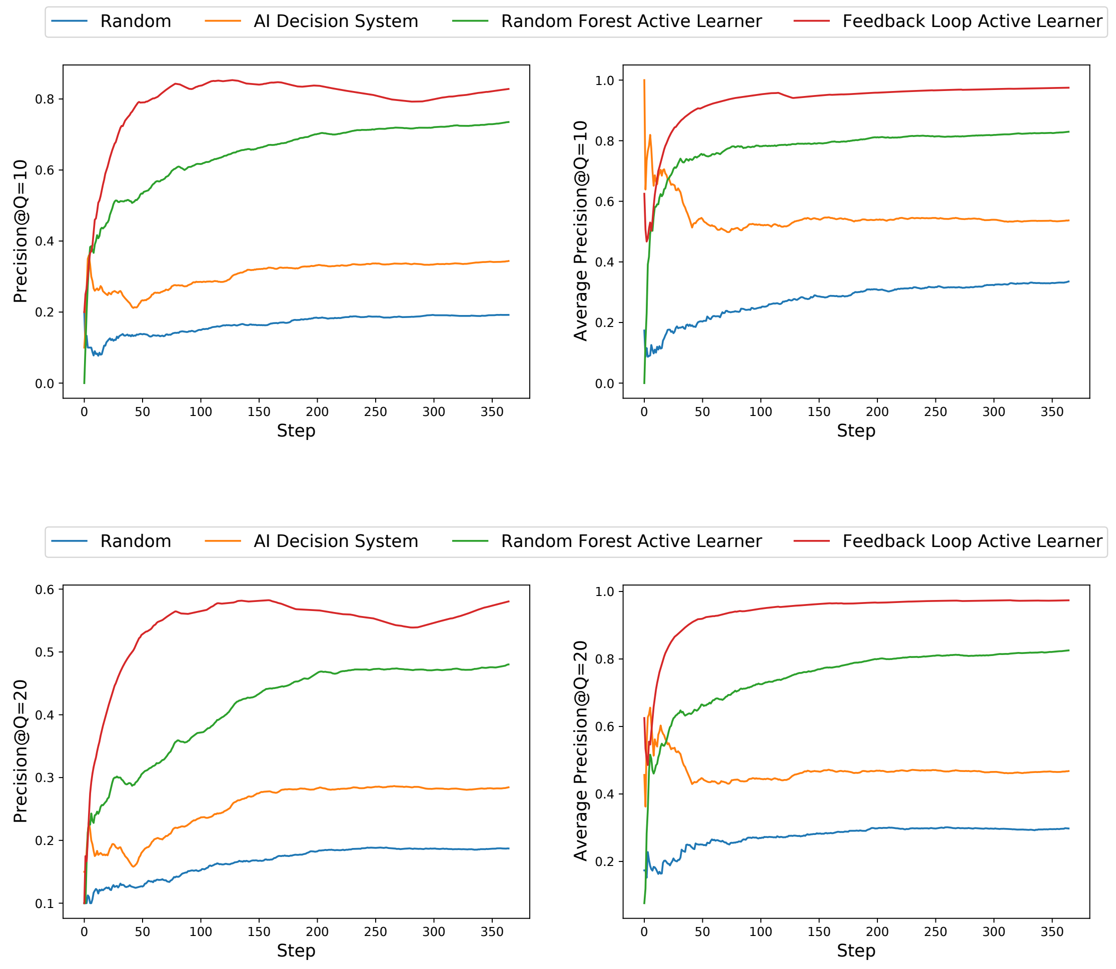

**Figure 6:** Test scores for Precision@Q and AveragePrecision@Q. FLAL outperforms the competing baselines in both metrics and with different $Q = \{10, 20\}$. FLAL's performance becomes strictly better in around 10-20 steps and converges around 50 steps. Another active learning mechanism, Random Forest Active Learner, also generates better performance compared to the original AI decision system performance. Random recommendation mechanism expectedly is the worst.

## 4.4. Different User Embedding Mapper

Figure 7 shows an ablation study on different user embedding mapper layers using precision@10 and precision@20. We compare identity, linear, nonlinear, and nonlinear-2 (2 nonlinear layers). The identity layer returns the same embedding space and the nonlinear layers use a sigmoid activation. The result suggests that one linear layer captures enough information as much as one nonlinear layer. On the other hand, the identity layer has gradually increasing performance for Precision@10, but with a slower convergence rate. Nonlinear-2 has the worst performance. The reason could be overfitting since the lack of data points.

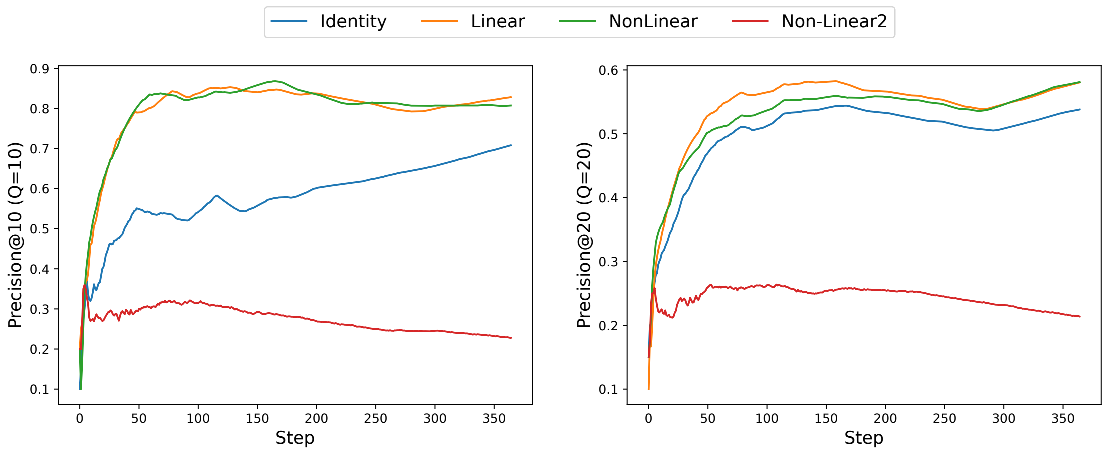

**Figure 7:** Ablation study of user embedding mapper. The linear and non-linear layers have competing performances at both metrics. The linear mapping converges slowly, and 2 non-linear layers suffer from the lack of available data points.

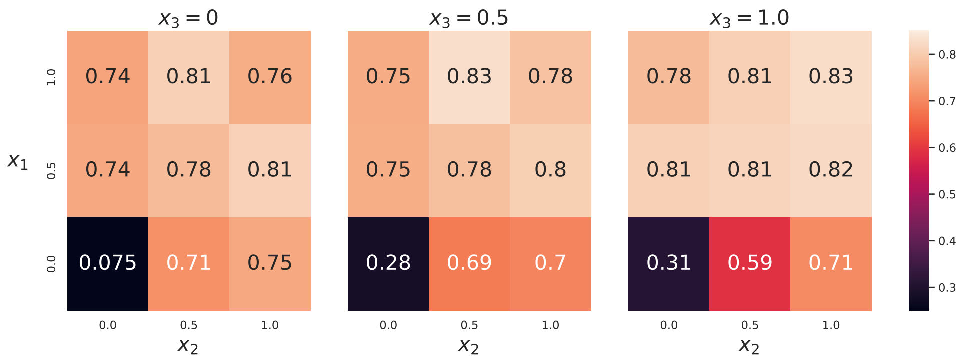

**Figure 8:** Sensitivity analysis of choosing $x_{1,2,3}$ for our loss terms on the Machine dataset. The reported blocks show an average of precision@10 across all steps of active learning. $x_1$ is the most effective term in our loss function as the absence of it generates much worse performance. $x_2$ and $x_3$ have similar effects since they aim to narrow between negative and positive samples.

## 4.5. Loss Function Term Sensitivity

Figure 8 shows a sensitivity analysis on objective function terms for the Machine dataset. Each block represents the last step cumulative average of precision@10 scores from FLAL parameterized by different $x_{1,2,3}$ values. All terms contribute to the performance, while the absence of the first term (i.e., $x_1 = 0$) makes the model performance very bad. This behavior is expected since the first term is the core part of the contrastive loss. The second term is not effective as the first term for the Machine dataset, but still increasing the value of $x_2$ makes the performance better. The third term has a similar effect as the second term. They both aim to narrow the positive and negative sample rankings. The best performance is achieved when hyperparameters are equal to 1, or $x_1 = 1$, $x_2 = 0.5$, and $x_3 = 0.5$.

## 5. Conclusion and Future Works

We investigate better and more effective ground truth generation by incorporating recommendation systems into AI decision systems and human collaboration for entity-based time-series data. We propose FLAL understanding the expertise and interest of a human over queries to make feedback more eligible and accurate using active learning. FLAL trains a personalized embedding mapper; uses features extraction and AI system decisions to solve the cold-start problem of recommender systems. FLAL performs better than competing baselines: random forest active learners, AI decision-based, and random recommenders; and it converges fast. Furthermore, our ablation studies show that the linear user embedding mapper is learning enough information and each term in the objective function contributes to the result.

In future work, we want to investigate this problem using different datasets and our proposed user simulation setting. We also desire to conduct human experiments to show the effectiveness of FLAL in real settings. Furthermore, we will optimize the number of recommendations instead of using it as the budget $Q$.

## Acknowledgments

The work was done when Mert Kosan was an intern at Visa Research.

## References

[1] K. Zheng, S. Cai, H. R. Chua, W. Wang, K. Y. Ngiam, B. C. Ooi, Tracer: A framework for facilitating accurate and interpretable analytics for high stakes applications, in: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, 2020, pp. 1747–1763.

[2] C. Rudin, Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, Nature Machine Intelligence 1 (2019) 206–215.

[3] M. Kosan, A. Silva, S. Medya, B. Uzzi, A. Singh, Event detection on dynamic graphs, arXiv preprint arXiv:2110.12148 (2021).

[4] M. Kosan, Z. Huang, S. Medya, S. Ranu, A. Singh, Global counterfactual explainer for graph neural networks, arXiv preprint arXiv:2210.11695 (2022).

[5] C. Chen, A. Seff, A. Kornhauser, J. Xiao, Deepdriving: Learning affordance for direct perception in autonomous driving, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 2722–2730.

[6] T. Stevens, Knowledge in the grey zone: Ai and cybersecurity, Digital War 1 (2020) 164–170.

[7] V. Lai, C. Chen, Q. V. Liao, A. Smith-Renner, C. Tan, Towards a science of human-ai decision making: a survey of empirical studies, arXiv preprint arXiv:2112.11471 (2021).

[8] X. Wu, L. Xiao, Y. Sun, J. Zhang, T. Ma, L. He, A survey of human-in-the-loop for machine learning, Future Generation Computer Systems (2022).

[9] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowledge-based systems 46 (2013) 109–132.

[10] S. Zhang, L. Yao, A. Sun, Y. Tay, Deep learning based recommender system: A survey and new perspectives, ACM Computing Surveys (CSUR) 52 (2019) 1–38.
[11] Y. Peng, A survey on modern recommendation system based on big data, arXiv preprint arXiv:2206.02631 (2022).
[12] R. Sabitha, S. Vaishnavi, S. Karthik, R. Bhavadharini, User interaction based recommender system using machine learning, Intelligent Automation and Soft Computing 31 (2022) 1037–1049.
[13] R. M. Monarch, Human-in-the-Loop Machine Learning: Active learning and annotation for human-centered AI, Simon and Schuster, 2021.
[14] Y. Liang, L. He, X. Anthony'Chen, Human-centered ai for medical imaging, Artificial Intelligence for Human Computer Interaction: A Modern Approach (2021) 539–570.
[15] H. Dong, V. Suárez-Paniagua, W. Whiteley, H. Wu, Explainable automated coding of clinical notes using hierarchical label-wise attention networks and label embedding initialisation, Journal of biomedical informatics 116 (2021) 103728.
[16] T. Dash, S. Chitlangia, A. Ahuja, A. Srinivasan, A review of some techniques for inclusion of domain-knowledge into deep neural networks, Scientific Reports 12 (2022) 1–15.
[17] Y. Kang, Y.-W. Chiu, M.-Y. Lin, F.-Y. Su, S.-T. Huang, Towards model-informed precision dosing with expert-in-the-loop machine learning, in: 2021 IEEE 22nd International Conference on Information Reuse and Integration for Data Science (IRI), IEEE, 2021, pp. 342–347.
[18] C. Chandler, P. W. Foltz, B. Elvevåg, Improving the applicability of ai for psychiatric applications through human-in-the-loop methodologies, Schizophrenia Bulletin (2022).
[19] Z. Liu, J. Wang, S. Gong, H. Lu, D. Tao, Deep reinforcement active learning for human-in-the-loop person re-identification, in: Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 6122–6131.
[20] S. Budd, E. C. Robinson, B. Kainz, A survey on active learning and human-in-the-loop deep learning for medical image analysis, Medical Image Analysis 71 (2021) 102062.
[21] G. Shani, A. Gunawardana, Evaluating recommendation systems, in: Recommender systems handbook, Springer, 2011, pp. 257–297.
[22] M. Zehlike, K. Yang, J. Stoyanovich, Fairness in ranking, part ii: Learning-to-rank and recommender systems, ACM Computing Surveys (CSUR) (2022).
[23] Y. Zhang, X. Chen, et al., Explainable recommendation: A survey and new perspectives, Foundations and Trends® in Information Retrieval 14 (2020) 1–101.
[24] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al., Wide & deep learning for recommender systems, in: Proceedings of the 1st workshop on deep learning for recommender systems, 2016, pp. 7–10.
[25] H. Wang, N. Wang, D.-Y. Yeung, Collaborative deep learning for recommender systems, in: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 1235–1244.
[26] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, P. Poupart, Representation learning for dynamic graphs: A survey., J. Mach. Learn. Res. 21 (2020) 1–73.
[27] E. Eldele, M. Ragab, Z. Chen, M. Wu, C. K. Kwoh, X. Li, C. Guan, Time-series representation learning via temporal and contextual contrasting, arXiv preprint arXiv:2106.14112 (2021).

[28] S. M. Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, S. Wu, C. Smyth, P. Poupart, M. Brubaker, Time2vec: Learning a vector representation of time, arXiv preprint arXiv:1907.05321 (2019).

[29] J.-Y. Franceschi, A. Dieuleveut, M. Jaggi, Unsupervised scalable representation learning for multivariate time series, Advances in neural information processing systems 32 (2019).

[30] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, C. Eickhoff, A transformer-based framework for multivariate time series representation learning, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2114–2124.

[31] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, D. Pei, Robust anomaly detection for multivariate time series through stochastic recurrent neural network, in: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 2828–2837.