The Value of Corrective Feedback in the Online Active Learning Paradigm

Mark Lindsey, *Member, IEEE*, Francis Kubala, *Member, IEEE*, and Richard M. Stern, *Fellow, IEEE*

***Abstract*—Online Active Learning (OAL) is a powerful tool for classifying evolving data streams using limited annotations from a human operator who is a domain expert. The objective of the OAL learning paradigm is to minimize jointly the classification error rate and the annotation cost across the data stream by posing periodic Active Learning (AL) queries. In this paper, this objective is extended to include identification of classifier errors by the expert during the typical workflow. To this end, Corrective Feedback (CF) is introduced as a second channel of interaction between the expert and the learning algorithm, complementary to the AL channel, that allows the algorithm to obtain additional training labels without disrupting the expert's workflow. Online Active Learning with Corrective Feedback (OAL-CF) is formally defined as a paradigm, and its efficacy is proven through experimental application to two binary classification tasks, Spoken Language Verification and Voice-Type Discrimination. Finally, the effects of adding CF to the OAL paradigm are analyzed in terms of classification performance, annotation cost, trends over time, and class balance of the collected training data. Overall, the addition of CF results in a 53% relative reduction in cost compared to OAL without CF.**

***Index Terms*—Machine learning paradigms, active learning, online learning, human-in-the-loop, binary classification, spoken language verification, voice-type discrimination.**

I. Introduction

In THE era of Large Language Models (LLMs) [2] and foundation models [3], [4], [5], the focus of the Machine Learning (ML) community has tended toward large neural networks trained on massive amounts of annotated data. This approach to ML is referred to as Passive Machine Learning (PML) [6], since the neural network is provided with a set of pre-determined training data rather than contributing as an active participant in the learning process. The PML training paradigm is illustrated in Fig. 1. Because of the high level of expressivity that comes with the billions of parameters contained in these neural architectures, as well as the vast array of data used to train them, PML-trained networks have been shown to perform well on a wide variety of tasks in many different environments, even without further updates to the classifier after deployment.

However, despite their popularity and the fact that they are highly effective in many use cases, these algorithms fail to generalize to conditions that are dissimilar from those seen in the training set. As such, these PML approaches fail to address certain real-world challenges of ML, including processing evolving data streams, adapting to unseen or unexpected variations in operational data, learning from limited labeled training data, handling extreme class sparsity and imbalance, and integrating user feedback. Fortunately, alternative learning paradigms, such as Online Active Learning (OAL), have been designed to address these challenges.

A. Online Active Learning

OAL addresses the practical issues of PML by introducing two fundamental changes to the learning paradigm: 1) the data are presented as a data stream rather than a static pool, and 2) the samples that are labeled and used for training are selected by the machine [1], [7]. So, in the OAL paradigm, an ML classifier is trained sequentially in *online* fashion, with the intention of adapting directly to the data stream on which the algorithm operates. This goal is achieved by allowing the algorithm to pose a limited number of *active* queries about samples observed in the data stream to a human operator who can provide accurate class labels. The classifier learns from these labels and subsequently classifies the unlabeled portion of the data stream. This process of querying, learning, and classifying is repeated throughout the data stream to give the algorithm a mechanism to adapt to distribution shifts in the data stream, or "concept drift" [8].

Since the data are presented in a stream in OAL, the annotated data pool available for training and adapting the classifier grows over time. This adaptation pool starts out as a small number of initialization samples, and then it grows as more data from the stream are annotated via active queries. Online processing presents a number of challenges that are not addressed by the PML paradigm. First and foremost, the volume of labeled data is much more limited, which makes it difficult to train a robust classifier. Second, data streams are not independent and identically distributed (i.i.d.), so the class imbalance becomes both more varied over time and more extreme than a static data pool. Finally, online training poses the question of which samples and how many samples should be annotated for training to achieve both the lowest error rate and the lowest annotation cost.

In the OAL paradigm, rather than relying on a data scientist to curate the training data, the machine becomes an active participant in the data selection process. An Active Learning (AL) sample selection strategy is employed in an attempt to determine which data points would be the most valuable to the training process if the associated labels were known. Then, the selected data are assigned labels by some oracle—in this paper,

Received 14 April 2025; revised 20 October 2025; accepted 30 November 2025. Recommended for acceptance by R. K. Iyer. *(Corresponding author: Mark Lindsey.)*

Mark Lindsey, Francis Kubala, and Richard M. Stern are with Carnegie Mellon University, Pittsburgh, PA 15213-3815 USA, and also with Probity, Inc., Herndon, VA 20170 USA (e-mail: mlindsey@probity.com).

Digital Object Identifier 10.1109/TPAMI.2025.3639522

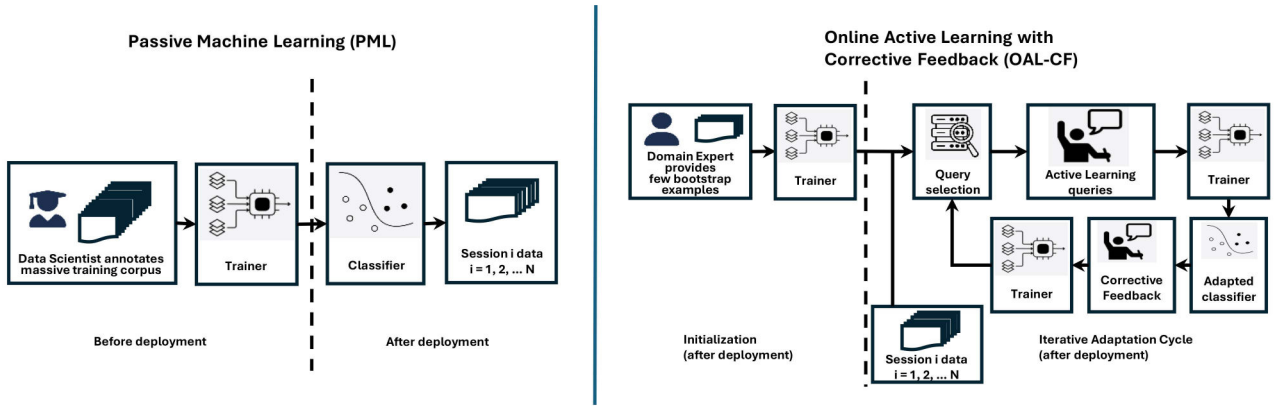


Fig. 1. Visual comparison of conventional PML to the OAL-CF paradigm.

a domain expert who also performs the task—after which the labeled data are made available to the classifier for training.

Existing OAL methods have been applied to a wide variety of tasks in both batch-based and sample-by-sample contexts [7]. Applications include spam email detection [9], text classification [10], object classification [11], object detection [12], and various other process monitoring tasks [13], [14], [15], [16]. In each application, AL query selection methods are applied to a streaming data problem to predict which samples are the most informative for training and adaptation. These methods are strictly limited to AL queries for obtaining user input, however. If the training process were allowed to be guided by active human input, the classifier could be better attuned to the particular needs of the user, even when training data is limited or the task is not formally defined before deployment.

B. Contributions

In this paper, we introduce *Corrective Feedback* (CF) as a second channel of interaction between the expert and the machine classifier in the OAL paradigm. We call this new formulation the Online Active Learning with Corrective Feedback (OAL-CF) paradigm (see Fig. 1). We also redefine and clarify the objective of OAL as a method for a *domain expert* (i.e., a human user with expertise in a specific task) to train a reliable ML tool that can aid him or herself without becoming a significant distraction from his or her normal workflow. In other words, the objective of OAL is to minimize jointly the classification error and annotation cost for a specialized task in a specific *operating environment*. Without the aid of a machine classifier, the expert's only recourse would be to review every sample in the data stream. Therefore, a machine classifier that reliably eliminates some of the uninformative samples in the stream can increase the expert's efficiency and reduce his or her cognitive burden.

In this paper, we show empirically that OAL-CF is a superior method for optimizing this objective, with a 53% relative reduction in cost compared to OAL without CF. We also provide detailed analysis of the behavior of OAL-CF to explain its success. In short, although it may be counterintuitive, the addition of CF has proven to lower the overall cost of annotation effort and error, as measured by the Interactive Machine Learning Metric (IMLM) described in Section IV-A. This is possible because the resulting error reduction from CF samples is significant enough that the additional validation effort becomes objectively worthwhile. Note that CF does not incur a hidden cost—every false positive error, including those that are later corrected by the expert, is counted equal to an AL query by the IMLM cost.

In the OAL-CF paradigm, CF is defined as the introduction of additional class labels obtained based on the expert's response to interaction with the classifier's predictions. Conceptually, CF is an important addition to the OAL pipeline because it provides a complementary information channel to AL. AL allows the machine to ask the expert questions; CF allows the expert to point out errors in the machine's predictions. Without both of these channels, communication between the machine and the expert becomes one-sided and even dangerously inefficient. For example, one commonly-cited challenge in online learning and AL algorithms is the classifier becoming overly confident in its own predictions [17]. This is especially dangerous when the relevant underlying statistics change slowly. If the classifier fails to detect these changes, it will make confident decisions that are incorrect.

CF is also powerful because, unlike AL, it does not require additional work beyond the expert's normal workflow. In any application where the expert performs downstream analysis or other processing using the output of the classifier, the classifier acts as a sieve that separates informative data from useless data. An imperfect classifier reduces the amount of useless data that need to be processed, but does not completely eliminate it. As such, a necessary part of the expert's workflow includes reviewing all target predictions to separate true positive predictions from false positives. CF takes advantage of this part of the workflow and obtains labels from this review process. AL queries, on the other hand, add extra steps to the expert's workflow.

The original motivation for this study was to find adaptable solutions for previously undefined tasks that involved detecting specific events in audio streams. Such tasks include, but are not limited to, detecting target audio in surveillance applications

where a human observing the audio stream is searching for information pertinent to potential threats or security risks. Considering this original motivation, the applications explored in this paper are defined to be binary detection or verification tasks in the audio domain where the detection or verification target is some specified rare audio event. Specifically, experiments and analysis are performed primarily on the Spoken Language Verification (SLV) task and later extended to the task of Voice-Type Discrimination (VTD). The goal in these applications, like the goal of OAL, is to perform the task as well as possible while simultaneously minimizing the number of annotations required for classifier training. As such, the success of the algorithms evaluated here will be determined with a set of evaluation metrics that consider both annotation cost and error rate.

The remainder of this paper is organized as follows. Section II reviews other extensions of OAL and existing learning paradigms that utilize human feedback. In Section III, we formally define the OAL-CF paradigm introduced in this paper. Section IV describes how the learning paradigms are evaluated, and Section V outlines the experimental configuration. We report experimental results and provide analysis of the results in Section VI. Limitations of the OAL-CF paradigms and our conclusions are summarized in Sections VII and VIII.

II. Related Literature

In this section, we review the related literature. This section is brief because prior work in this area has been quite limited. We begin by covering other significant ways that the OAL paradigm has been adjusted or extended, and then proceed to existing methods that incorporate some form of CF.

A. Extensions of OAL

OAL has been extended in various ways to address specific challenges. One challenge that has received significant attention is that of drifting or time-varying data streams. Even with expert guidance, OAL systems require additional functionality to navigate the concept drift that is inherent in most data streams. Concept drift is often handled by introducing some form of Drift Detection Method (DDM) [18] and adjusting the OAL algorithm in some way when the DDM is triggered. Some approaches allocate additional AL queries to the sections of the stream where drift is more likely [19], [20], some start training a new classifier that will eventually replace the pre-drift classifier [21], and others learn to weight the predictions made by a dynamic classifier with predictions made by a stable classifier [22].

Another challenge that is particularly salient in OAL is that of imbalanced class representation. This common ML problem is exacerbated by nature of classes not being independent and identically distributed in real-world data streams, as well as the extreme levels of class sparsity in many data streams [23]. Some methods address class imbalance in a manner similar to many offline algorithms, by oversampling the minority class or undersampling the majority class [24]. Other methods work more directly with the classification process by learning a low-dimensional projection that makes discrimination easier for the minority class [25], or by utilizing a loss function with non-uniform weightings to compensate for the imbalance [26].

Finally, some OAL algorithms account for the realistic possibility that the responses to AL queries may not always be clear, accurate, or timely. For example, many algorithms based on "expert advice" must parse through differing annotations from multiple experts to determine how to incorporate the annotations [27]. Other algorithms consider the annotator to be only "mostly reliable", and, as such, develop methods that account for occasional label noise [28]. Yet other algorithms address the case where annotations are delayed so that algorithm cannot adapt immediately [29].

B. Algorithms That Utilize Corrective Feedback

Many Human-In-The-Loop (HITL) algorithms have been developed over the years, but only some of these algorithms utilize the human in the loop explicitly as source of CF [30]. In many cases, the human acts as an active selector and annotator of important samples before training [31], and in some cases the sole purpose of the human is to guide training manually [32]. The few algorithms that allow a human operator to correct classification errors after predictions have been made include some Interactive Machine Learning (IML) and Reinforcement Learning (RL) algorithms.

IML is broadly defined as "an interaction paradigm in which a user or user group iteratively builds and refines a mathematical model to describe a concept through iterative cycles of input and review." [32]. As such, many IML algorithms focus more on the user interface than the learning algorithm or CF [33], [34], [35]. IML algorithms that do feature CF take various forms. Some allow explicit manipulation of the classifier's confusion matrix by the user to achieve optimal behavior [36]. Others allow control over the data and annotation processes [37] or feature selection [38] to indirectly manipulate the behavior of the system.

RL algorithms actively interact with their operating environments in order to optimize some given reward function. CF has been used in RL applications as a way for a human in the loop to play an active role in teaching the machine a particular behavior, and as a method of stabilizing the learning process [39], [40], [41]. Inverse Reinforcement Learning (IRL) algorithms additionally learn the reward function itself by observing demonstrations of a task being performed [42]. IRL is often used in Apprenticeship Learning and Imitation Learning applications, which allow corrective feedback in the form of additional demonstration [43].

III. Online Learning Paradigm Definitions

In most ML classification algorithms, a classifier is trained by being presented with many examples comprised of pairs of features ($\mathbf{x}$) and labels (y). Through some mathematical process, the classifier learns the mapping of features to labels, $f(\mathbf{x}) = y$. A critical assumption of these Machine Learning algorithms is that the features extracted from the training data are from the same distribution as the operational data. All three learning paradigms (PML, OAL, and OAL-CF) explored in this paper

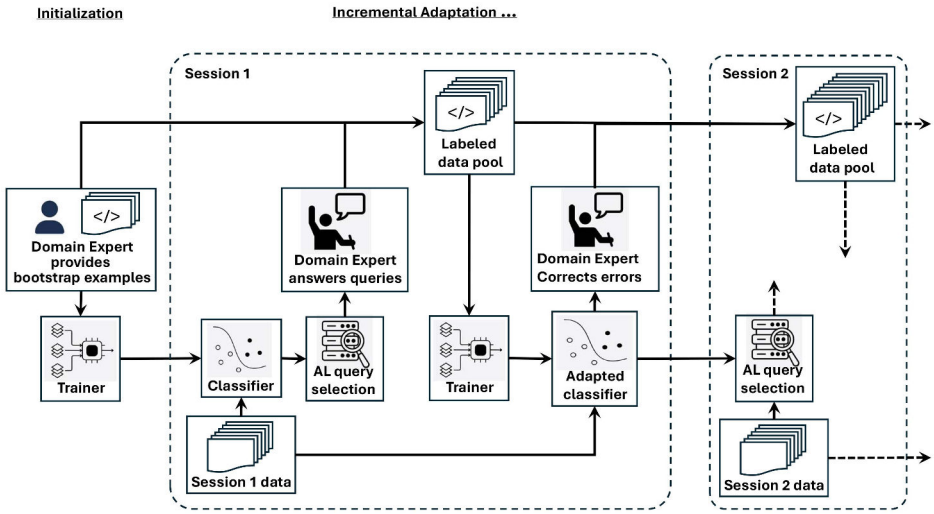


Fig. 2. Overview of the OAL-CF paradigm. When the CF block is removed, this becomes the OAL paradigm (without CF). For both paradigms, the classifier is first initialized with a small bootstrap corpus and then introduced to the data stream. Within each online session, the expert annotates a small number of AL samples, the classifier is updated, and class predictions are made for all current session data. CF is provided for the predictions in the OAL-CF paradigm, and the labels derived from CF are added to the training pool for the next session.

train classifiers in this way; the difference lies in the way the examples are collected and presented. These differences are explained below as OAL and OAL-CF are explained in detail. Reference code from the first author's doctoral thesis [1] is available on GitHub[1].

A. OAL Definition

As described above, OAL is a machine learning paradigm that trains a classifier on a data stream. The classifier learns from samples obtained directly from the data stream in the form of periodic active queries posed by the machine to the user for annotation. From this definition, we see that OAL differs fundamentally from PML in that is is both *online* and *active*.

The OAL process begins when the classifier is initially trained on a small *bootstrap corpus* of target and non-target samples prepared by a domain expert, who is also the operational user. In this work, the bootstrap corpus is defined to contain only four samples from each class, and it comes from an environment that differs from the target environment in the data stream. The bootstrap corpus data then become the first samples added to the *adaptation pool*, the growing reserve of annotated samples used to adapt the classifier.

After initialization, one batch of the data stream (defined as 720 samples or approximately one hour of audio in this paper) is presented at a time to the algorithm, beginning a *session* of processing. Within each session, the OAL algorithm completes three steps:

1) *AL queries:* A small sample of data from the current batch is selected by the machine, labeled by the expert, and added to the adaptation pool.
2) *Adaptation:* The classifier is adapted by fine-tuning on all labeled data available in the adaptation pool.
3) *Prediction:* The classifier makes predictions about each sample in the session. The predictions are presented to the expert.

After these three steps, the next batch of data is loaded from the stream and the next session begins. This process is repeated for all data in the stream.

B. OAL-CF Definition

OAL-CF, like OAL, is also an online algorithm that actively identifies informative samples for labeling (see Fig. 2). The important difference between OAL and OAL-CF is the addition of CF.

Conceptually, CF is crucial to making OAL efficient and reliable in real operation. CF takes advantage of the expected workflow of the expert by obtaining labels from downstream analysis on the classifier's predictions. Since the expert will analyze these samples regardless of whether or not the algorithm collects CF, additional ground-truth class information can be gleaned from their analysis without adding any additional task to the expert's workflow. So, compared to AL queries, CF acquisition takes advantage of an existing part of the expert's workflow, but CF annotations can only be applied to sessions after the session in which the annotations were acquired.

In terms of technical algorithmic details, the addition of CF means two additional steps after the predictions are reviewed by the expert. The first additional step is to acquire CF labels. We make the assumption here that the expert will only review and classify samples that the machine has predicted to be from the target class. This means that all positive predictions (i.e., false positives and true positives) in the session get labels. CF is defined in this way because it aligns with the expert's workflow, as target samples contain information that will be used in downstream tasks, and all non-target samples are considered uninformative and will be thrown out. Additionally, there is no way to find false negative errors other than listening to the entire

[1] https://github.com/markrl/oalcf

data stream, which is the baseline cost of manually looking for the target class without the help of technology. The model is strongly dependent on the already learned samples, in much the same way that speaker verification separates the target from potential impostors (non-targets), which is the primary purpose of CF.

The second additional step that comes with CF is to adapt the classifier again after the CF samples have been added to the adaptation pool. This is necessary because the classifier should be up-to-date before AL queries are selected in the next session.

IV. Evaluation Methodology

This section describes the metrics and pipeline used to evaluate the learning paradigms explored in this paper. This section is important because of the inherent difficulty of evaluating online HITL algorithms, and because the chosen evaluation method here is different from the methods commonly used in other work. The primary challenge of evaluating these algorithms stems from the need to account for both the labor required to obtain annotations for classifier training and the error rates incurred by the classifier simultaneously. Furthermore, it is difficult to compare the online learning paradigms discussed in this paper because the subset of samples used for training varies from experiment to experiment. The methods described below are designed to address these challenges.

A. Evaluation Metrics

To address the unique challenges of evaluating HITL algorithms, we employ the Interactive Machine Learning Metric (IMLM) as the primary evaluation metric in this paper. We also track the Detection Cost Function (DCF), False Negative Rate (FNR), and False Positive Rate (FPR) as secondary diagnostic metrics that may be more familiar to the reader.

The IMLM is key to the fair comparison of different learning paradigms, since it considers annotation cost in addition to error cost [44]. In order to combine these two types of costs across modalities (i.e., number of annotations made and error rates), both are converted to units of time. For this conversion, it is assumed that a single annotation will cost approximately the same amount of time as it would to review a prediction from the machine and classify it as a false positive. False negatives cost more time than false positives or annotations because they are more rare and more damaging to the detection or verification operation. They are also unlikely to be caught by the domain expert who is only attending to the samples predicted by the machine to contain the target.

As such, false positive classifications and annotations are assigned the same cost, while false negatives have a higher cost in (1). Here, N is the total number of samples in the data stream, N_{ann} is the total number of explicit annotations made by the expert (i.e., the number of bootstrap samples, AL queries, and false positives), and N_{fn}, N_{fp}, and N_{target} are false negatives, false positives, and target samples, respectively. Note that, because the denominator associated with false negatives is less than the denominator of the false positives and annotations (i.e., $N_{\text{target}} \ll N$), the cost of false negative errors is relatively amplified. In our work we assume that the level of amplification is inversely proportional to the prevalence of the target class [44].

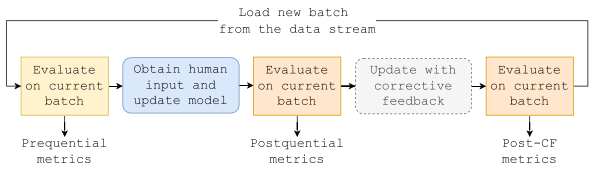


Fig. 3. The order of prequential evaluation and postquential evaluation in the pipeline of the OAL-CF paradigm.

$$\text{IMLM} = \frac{N_{\text{ann}} + N_{\text{fp}}}{N} + \frac{N_{\text{fn}}}{N_{\text{target}}}. \tag{1}$$

IMLM is tracked across all sessions and reported as a single number calculated from the accumulated errors and annotations across the entire run. IMLM can be understood intuitively, as all values less than 1.0 are more time efficient than manual analysis, and all values greater than or equal to 1.0 are equally or less efficient compared to manual analysis.

DCF is a standard NIST metric for evaluating performance on detection and verification tasks [45]. It is a weighted combination of FNR and FPR where FNR is weighted more heavily than FPR. This formulation ensures that false negatives, which are more detrimental but less common than false positives because of the relative scarcity of the target class, are not overlooked. This makes DCF advantageous compared to typical classification evaluation metrics, such as accuracy, F-measure, Equal Error Rate (EER), or Area Under the Receiver Operating Characteristic Curve (AUC), that do not consider the scenario where the target class is more important and often more rare than the non-target class, and they do not force the algorithm to chose an operational threshold.. The standard DCF [46] calculation used in this paper is shown in (2), where P_{fn} and P_{fp} are FNR and FPR, respectively, and $\xi = 0.75$.

$$\text{DCF} = (1 - \xi)P_{\text{fp}} + \xi P_{\text{fn}} \tag{2}$$

While DCF, FNR, and FPR are informative, none of these measures captures the annotation cost for the expert. Thus, these three measures are secondary to IMLM. DCF, FNR, and FPR are also tracked across all sessions and reported as a single cumulative number.

B. Evaluation Pipeline

Online HITL algorithms like OAL are typically evaluated prequentially [47], meaning that evaluation is done on the full batch of new data immediately after it is received, before the classifier has adapted to the new batch. This allows evaluation to be done on the entire data stream before the classifier has seen any ground truth labels for any given batch, making comparison across algorithms fair and simple. However, prequential evaluation does not reflect how well the classifier adapted to the current session, neither does it paint an accurate picture of what the final prediction of the classifier would be after adaptation.

As such, we employ postquential evaluation after the adaptation step (see Fig. 3) in order to capture a more realistic representation of the classifier's performance. The term "postquential"

appears in only a single piece of work in the literature [48] prior to its recent formal introduction in our own work [44]. Postquential evaluation has likely received limited attention in the past because many factors (e.g., model choice, initialization, random seed, learning parameters, etc.) can lead to varying selections of which data points are annotated. These variations, in turn, lead to different effective training and evaluation sets, making it difficult to compare results across experiments that are exposed to different unlabeled test samples. We circumvent this issue by tallying costs for both errors and annotation labor for all samples from the data stream, including samples that were annotated or corrected by the expert, and then evaluating the total cost with IMLM.

Traditionally, evaluating all samples including those that have been annotated might be seen as "testing on the training data" since the classifier will be exposed to annotated test samples during training. However, we assert that, for training paradigms like OAL, querying the expert is an integral part of the algorithm and a method for obtaining data labels that is equally as viable as algorithmic prediction, albeit more expensive. Under this assertion, and since IMLM reflects the cost of querying the oracle, IMLM can be used to compare two OAL runs with different query budgets and query sample selections and, therefore, with different subsets of unseen samples from the data stream. This aspect of IMLM is essential for evaluating online HITL learning algorithms.

In addition to prequential and postquential evaluation metrics, we also provide evaluation metrics calculated after the classifier has been updated with the CF samples from the current session. These Post-CF metrics serve as indicators of whether the classifier truly absorbs the information provided by CF. These metrics also begin to indicate the rate at which the error would decline if the CF step were to be repeated multiple times within the same session.

V. Experimental Configuration

This section outlines the configuration of the experiments reported in this paper. The goal of the experiments is to highlight the advantages of OAL-CF, and to provide a view into the inner workings of the paradigm for further analysis. As such, the experiments basically consist of running PML, OAL, and OAL-CF paradigms on real data, observing their behavior, and comparing their performance. We also perform additional experiments which control the two following independent variables:

1) AL query budget allotment
2) Data stream order of presentation

Primary experiments and analysis are performed on the SLV task, but we also apply the paradigms to a secondary task, VTD, to verify our findings. The tasks and corpora, classifier architecture, and learning parameters for the experiments are discussed below.

A. Tasks and Corpora

The tasks used to evaluate the OAL-CF paradigm, along with the associated corpora, are presented here. There are three unique requirements for the organization of the data processed by an OAL algorithm. First, the volume of data being evaluated should be sufficiently large so that it can be broken down into batches of reasonable size. Second, the data should have some consistency in the way it was recorded—either by a single microphone in a fixed location, or by a common audio collection system. Third, the data should be presented in a set order, preferably in the order it was collected if such information is available. The first dataset discussed below, for the SLV task, is organized so that it also fits the criteria. The second, intended for the VTD task, meets all of these criteria without requiring further organization.

TABLE I
SLV Corpora Split Sizes in Terms of Number of Utterances

Split	AC	CR	SEA	SA
Train	982,800	666,720	1,363,680	797,040
Val	51,840	112,719	112,320	66,240
Test	56,880	114,480	90,720	68,400

1) Spoken Language Verification: SLV is the binary classification task of determining whether the speech in an audio clip contains a specified target language. This is different from the multi-class classification task of Language Identification (LID) where the objective is to predict which language out of a list of enrolled languages is being spoken in a given utterance [49]. SLV is also different from language detection, which involves detecting a target language in an audio stream with potential overlap from other languages. A closer analogue to SLV would be Automatic Speaker Verification (ASV), which, while focusing on the speaker's identity rather than the language he or she is speaking, simply discriminates utterances produced by the target speaker from utterances from other individuals. Note that ASV, despite not being defined as a detection task, also uses an evaluation metric similar to DCF [50].

Data for the SLV task come from Mozilla's Common Voice Corpus [51]. Common Voice is a large speech corpus with ongoing data collection from all over the world. Among many other types of metadata, the annotations for Common Voice denote the language spoken in each utterance and the location where the utterance was recorded. Using these annotations, the data can be sorted into corpora according to geographic region. Here we use corpora containing utterances from languages found in four regions: African Continent (AC), Caucasus Region (CR), Southeast Asia (SEA), and South Asia (SA). All four corpora are divided into training, validation, and test splits according to the splits defined in the Common Voice corpus. Experimental results are recorded for the test splits only, but the training split is used for bootstrap corpus formation and PML training, and the validation split is used for PML validation. The corpus sizes are shown in Table I.

In an effort to investigate rare target classes, target languages were chosen to be the languages with approximately 1% to 5% prevalence in their respective corpora. The class breakdown for the test splits of the four corpora can be found in Table II. Each language has an associated abbreviation that is displayed in the tables.

TABLE II
PREVALENCE OF ALL TARGET LANGUAGES IN THE SLV CORPORA

Language	Abbr.	Corpus (duration)	% of test split
Basaa	bas	AC (82 hours)	0.95
Hausa	ha		1.16
Yoruba	yo		1.75
Armenian	hy-AM	CR (162 hours)	3.74
Central Kurdish	ckb		4.60
Chuvash	cv		1.13
Kurmanji Kurdish	kmr		3.41
Kyrgyz	ky		1.41
Serbian	sr		1.34
Tatar	tt		4.33
Cantonese	yue	SEA (146 hours)	2.86
Indonesian	id		3.97
Vietnamese	vi		1.39
Hakha Chin	cnh	SA (119 hours)	1.11
Dhivehi	dv		3.22
Hindi	hi		4.59
Malayam	ml		1.03
Marathi	mr		2.55
Odia	or		1.47
Saraiki	skr		1.47

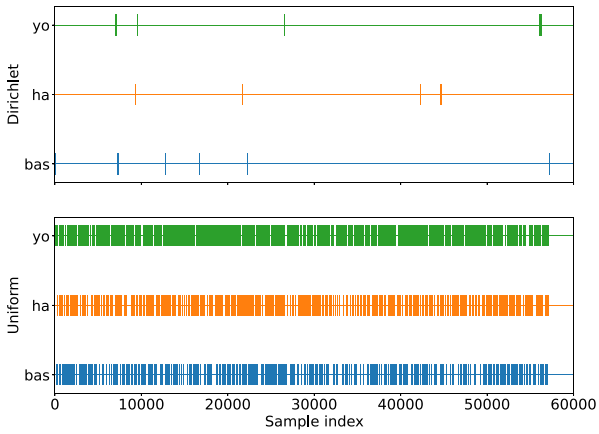


Fig. 4. Different orderings of the AC-SLV Corpus test split: (top) Dirichlet ordering, and (bottom) uniform ordering. Note that the samples are evenly dispersed in the uniform ordering and clustered into larger groups in the Dirichlet ordering.

Data from Common Voice are not intended for online tasks, and, thus, do not have a pre-defined order in which the samples are presented. To fit the requirements for datasets in this work, we define a fixed ordering for these corpora. Inspired by existing work in the Test-Time Adaptation field, we determine this ordering using a Dirichlet distribution [52]. In contrast to an i.i.d. distribution, which would mix samples of all classes together uniformly, the Dirichlet distribution simulates a non-i.i.d. distribution that exhibits class clustering in time. This type of class clustering is typical of real-world data (e.g., the VTD corpora), makes the task more challenging, and serves as a unique aspect of online learning that can be exploited for later improvements. The difference between i.i.d. and non-i.i.d. ordering can be observed in the visualization of class distribution shown in Fig. 4.

2) Voice-Type Discrimination: VTD is the relatively unexplored task of identifying regions in audio recordings that contain live speech. *Live speech* is defined here as speech produced spontaneously in physical proximity to the recording device. This means that *media speech*—pre-recorded or broadcast speech that is played through a loudspeaker into the recording device—is not considered to be live speech [53].

TABLE III
LIST OF SELECTED MICROPHONES AND THE ASSOCIATED TARGET PREVALENCE FOR EACH ROOM OF THE VTD TASK

Corpus	Room name	Microphone	% target prevalence
SRI	Room 1	Mic 20	11.62
	Room 2	Mic 16	12.48
	Room 3	Mic 16	14.69
	Room 4	Mic 20	8.80
LB	Apartment	Mic 19	24.76
	Hotel	Mic 19	23.47
	Office	Mic 13	25.02

For VTD, the detection target is composed of all samples containing live speech, including samples that contain both live speech and media speech. Any sample that does not contain live speech is considered to be non-target. Samples are pre-defined as abutting segments of audio, five seconds in length. An operating environment for the VTD task is a particular microphone. A session is defined to be one hour long, which means that there are 720 samples per session. The order in which the samples are presented to the online learning algorithms is consistent with the temporal order in which the samples appear in the recordings.

Data for the VTD task come from two corpora: the SRI Corpus (collected by SRI International) and the Lionbridge (LB) Corpus. These two corpora will be made public in coming months. The SRI Corpus is composed of recordings made in four rooms, recorded from five microphone-pair locations in each room, resulting in 20 operating environments. The LB Corpus comes from recordings made in three rooms using seven microphone locations in each room, resulting in 21 operating environments. Between the two corpora, there are 4,583 hours of data split into 41 environments of varying length. The corpora contain 896 hours of live speech audio produced by 75 unique speakers in English, Mandarin Chinese, Spanish, and Russian.

For the purposes of this work, only one microphone was used for each room, resulting in seven different environments. The microphone for each room was intentionally chosen to be the most disadvantaged microphone—behind furniture or walls, far from areas where conversations might occur, etc. Table III shows which microphones were used in each room for reference.

Target samples are relatively uncommon in all of these sessions, with 18.85% target class prevalence overall. Class distribution for each individual environment is shown in Table III. The rooms in the LB corpus typically contain more target samples than the SRI corpus. We also note that target samples are not distributed uniformly throughout the data stream, but rather clustered together in time. This sometimes results in entire sessions without any target occurrences while other sessions have relatively high target prevalence.

B. Input Features

The two application tasks in this paper use different features as input to the system, all of which are derived from pre-trained embedding models. For the purpose of this paper, the embedding

models were frozen and not trained end-to-end with the classifier network. We note that, even though these embedding models are sequence-to-sequence models, the output sequence is reduced to a single vector by calculating the mean vector over the sequence.

For the SLV task, the feature used is embeddings from an ECAPA-TDNN network [49], [54] trained to perform Spoken Language Identification on the VoxLingua107 dataset [55]. This embedding model comes from SpeechBrain[2]. These features only use 256-dimensional vectors to represent each sample. VoxLingua107 includes 9 of the 20 target languages specified in Section V-A1: Hausa, Yoruba, Serbian, Tatar, Indonesian, Vietnamese, Hindi, Malayalam, and Marathi.

For VTD, the feature of choice is a concatenation of x-vectors and WavLM embeddings. X-vectors, which are a form of speaker embedding [56], are utilized to exploit the realistic nature of the VTD corpora—since the live speech being detected usually comes from the same few individuals, detecting the presence of speech from these individuals is very useful. WavLM is a more generic representation of speech that has been shown to perform well on a large number of speech tasks [57]. Both types of embeddings are extracted using pre-trained embedding models from SpeechBrain[3] and Hugging Face[4], respectively. The x-vector embeddings have 512 dimensions, and the WavLM embeddings have 1,024 dimensions, so the total number of dimensions per sample is 1,536.

It was found empirically that including context in the input features improved classifier performance significantly for these tasks. As such, in this paper, features with context are generated by concatenating the features for the sample in question with the features for the 3 samples prior and the 3 samples following, a total of 7 samples worth of features. This large concatenated feature is used only to classify the sample in question in the center of the feature vector. This concatenation method is used for both training and inference in all learning paradigms.

C. Classifier Architecture

The classifier used in OAL paradigms should be lightweight, easy to adapt, and capable of achieving a very low error in the operating environment given sufficient adaptation data. In this paper, the chosen model that fits all these requirements is the neural network depicted in Fig. 5. This architecture is composed of a contrastive network and a classifier network. The contrastive network is a Siamese network that projects the input features into two distinct clusters using a contrastive loss [58] based on cosine distance, $\mathcal{L}_{\text{contrast}}$. The contrastive loss is calculated based on pairs of training samples using (3), where $y = 1$ when the two points being compared come from the same class, and $y = 0$ when they come from different classes; $D_{cos} \in [0, 1]$ is the cosine distance between the two points; and $m = 1$ is the margin parameter.

$$\mathcal{L}_{\text{contrast}} = yD_{cos}^2 + (1 - y)(m - D_{cos})^2 \tag{3}$$

[2]https://huggingface.co/speechbrain/lang-id-voxlingua107-ecapa
[3]https://huggingface.co/speechbrain/spkrec-xvect-voxceleb
[4]https://huggingface.co/microsoft/wavlm-large

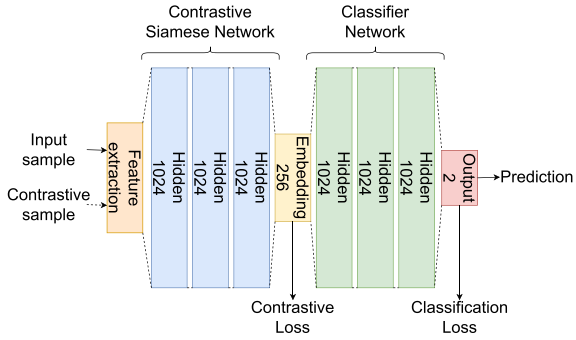


Fig. 5. The contrastive classifier architecture used in all experiments.

TABLE IV
DEFAULT SHARED LEARNING PARAMETERS FOR ALL EXPERIMENTS

Parameter(s)	Value/Setting
Optimizer	Adam
Learning rate	10^{-4}
Weight decay	10^{-5}
Batch size	256
Early stopping patience	15 epochs
Early stopping margin	10^{-3}

The classifier network then classifies the projected features into the target and non-target classes. The classifier portion is trained with a weighted negative log likelihood loss, $\mathcal{L}_{\text{class}}$, which defines the cost of each class based on Inverse Class Frequency (ICF) [59]. The ICF weighting formulation is shown in (4), where w_y is the weight of class y, N is the total number of training samples, N_y is the number of samples that belong to class y, and C is the total number of classes (in the case of binary classification, $C = 2$). To emphasize the importance of target samples, the ICF weight for the target class is multiplied by an additional factor of 10.

$$w_y = \frac{N}{C \cdot N_y} \tag{4}$$

The contrastive loss and classifier loss are combined into a total loss as a linear combination with mixing parameter ψ, as shown in (5). It was found empirically that setting the mixing parameter to $\psi = 0.09$ works well for these experiments.

$$\mathcal{L} = \psi\mathcal{L}_{\text{class}} + \mathcal{L}_{\text{contrast}} \tag{5}$$

D. Learning Parameters

Many of the learning parameters are shared across the three paradigms in this paper. Table IV shows the parameters that are common across all three paradigms.

Unless otherwise specified, both online paradigms use a default AL query budget of 8 samples per session. The bootstrap corpus is always composed of 8 samples, 4 from each class. Each session consists of 720 samples, which represents one hour of audio when samples are 5 seconds apiece on average.

For simplicity, there is no memory limitation for any paradigm. This means that the online paradigms always have access to all samples that have been added to the labeled adaptation pool.

TABLE V
PRIMARY SLV PERFORMANCE RESULTS FOR EACH PARADIGM. OAL-CF ACHIEVES THE BEST PERFORMANCE USING NEARLY ALL METRICS.

Paradigm	IMLM	DCF	FNR	FPR
PML	10.286	0.189	0.252	**0.001**
OAL	0.105	0.069	0.091	0.002
OAL-CF	**0.049**	**0.027**	**0.035**	0.002

It should also be mentioned that, because of the large number of training cycles required of the two online paradigms, it is not uncommon for denormal numbers to appear in the model parameters. To avoid the ill effects of denormal numbers, we simply set all parameters with denormal values to 0 (using `torch.set_flush_denormal(True)`).

VI. RESULTS AND ANALYSIS

In this section, we present the results of the SLV and VTD experiments outlined in the previous section. We begin with the primary experiment, which compares the three paradigms for the SLV task. Then we perform a series of analyses and demonstrate the performance of these paradigms on the VTD task.

A. Paradigm Performance Comparison

The primary SLV results are presented in Table V. These scores represent the postquential performance of each paradigm. In every category other than FPR, the OAL-CF paradigm performs the best, followed by the OAL paradigm and then the PML paradigm. Statistical tests (discussed in depth in Section VI-G) indicate that, in terms of IMLM, DCF, and FNR, OAL is significantly better than PML, and OAL-CF is significantly better than OAL. PML indeed achieves a marginally lower FPR than the online paradigms, but it does so at the cost of rejecting a large volume of valuable target samples.

We note that the IMLM of the PML paradigm is quite high. This is because the number of annotations, N_{ann}, used to train and validate the network in the PML framework is very large. It is also worth noting that, even if those annotations from outside the data stream were not considered (i.e., we assume $N_{\text{ann}} = 0$), the IMLM would still be 0.253, much worse than either of the online paradigms. The other paradigms are much more efficient with regard to IMLM because they are designed to adapt to the data stream by using a small number of examples taken directly from the operating environment. This ameliorates the common problem of mismatch between test and training data. The addition of CF makes OAL-CF more efficient than OAL in terms of overall cost of annotation and errors.

B. Impact of the Two Interaction Channels

The primary experiment makes it clear that the addition of CF is beneficial to the OAL paradigm. Further experimentation with different AL budget allocations elucidates how the two interaction channels, AL and CF, work when both are present. Fig. 6 shows the IMLM and DCF achieved by the two paradigms with increasingly large AL budgets, ranging from 2 to 24 AL samples per session.

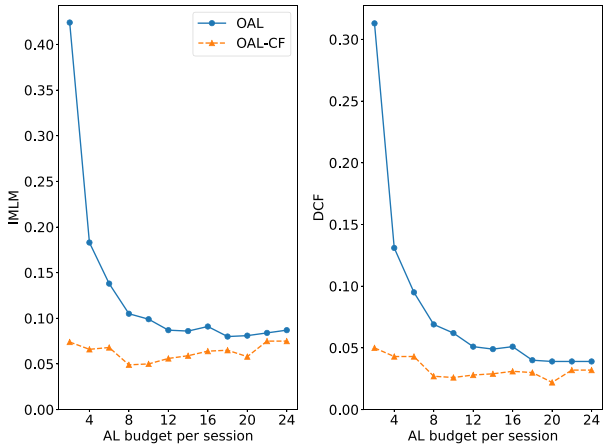


Fig. 6. OAL and OAL-CF performance in terms of IMLM and DCF given a range of AL budgets. IMLM is minimized at 18 samples per session for OAL and 8 samples for OAL-CF. OAL-CF consistently outperforms OAL.

First, we note that the results from OAL-CF are better than those from OAL for all AL query budgets. In terms of IMLM specifically, the worst IMLM result for OAL-CF is lower than the best IMLM result for OAL. Such dominant performance further confirms the importance of adding CF. Second, both curves have a similar trajectory, where increasing the query budget yields lower IMLM and DCF to a certain point, after which the improvements cease. While the gap between the paradigms narrows with increasing query budgets, OAL-CF can achieve much better performance than OAL with very few AL queries. Third, we see that the IMLM and DCF of the OAL paradigm are minimized when the AL budget is set to 18 queries per session. For OAL-CF, we see that DCF is lowest at 20 queries per session, but IMLM indicates that a budget of 8 queries is superior because it achieves a low error rate using fewer annotations. This highlights yet another advantage of OAL-CF—performance is optimized with relatively little extra required annotation effort.

C. Trend Analysis

Here we analyze the cost accumulated as a function of time for the two online paradigms. The plots in this section display the metrics cumulatively over time, meaning that at each point on the curve represents the IMLM or DCF from all errors and annotations for all languages up to and including the associated session indicated on the horizontal axis. Note that, since some corpora are shorter than others, the languages from the shorter corpora only add to the cumulative metrics until their last session. For example, languages from the AC-SLV corpus are 79 sessions in length, so they only contribute to the cumulative error and annotation counts for the first 79 sessions.

In this paper, we primarily consider the postquential scores of the classifier. The top panel of Fig. 7 shows the postquential IMLM for OAL and OAL-CF. We can see from this plot that the two paradigms start near the same place and experience the same inconsistent downward trend. This initial inconsistency can be understood as a minimally initialized classifier struggling to adapt to the environment. However, the OAL-CF curve trends

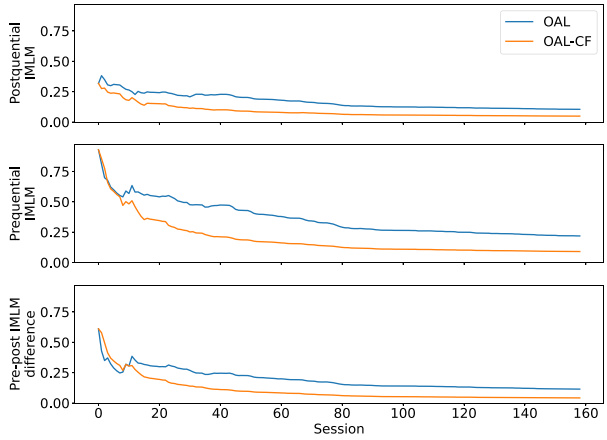


Fig. 7. Trends across time for postquential IMLM (top), prequential IMLM (middle), and the difference between pre- and postquential IMLM (bottom). OAL-CF consistently achieves better results than OAL.

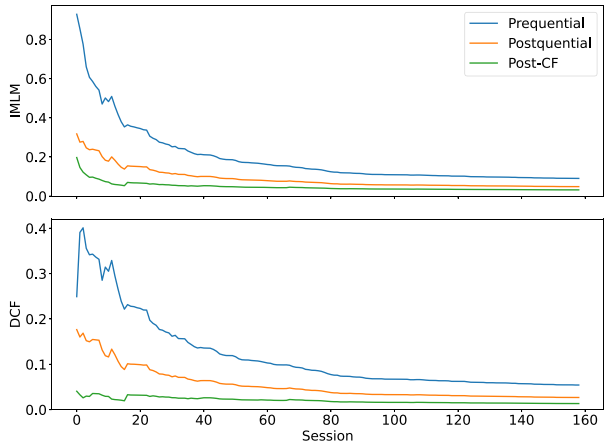


Fig. 8. IMLM and DCF trends across time for prequential, postquential, and post-CF evaluation of OAL-CF. As expected, results improve as more relevant training data becomes available.

downward much more quickly than OAL, maintaining notably lower IMLM for the entirety of the run. Since the only difference between these paradigms is the presence or absence of CF, it is clear that additional annotations from CF give OAL-CF the advantage.

The middle panel of Fig. 7 shows the same trends for prequential evaluation. The prequential trends are important because they represent the traditional method of evaluating online algorithms [47], and because they isolate the impact of the annotations gathered in previous sessions from the annotations acquired via AL queries for the current session. In contrast to what we observe in the postquential trend plots, OAL tracks OAL-CF very closely until session 10, indicating that the limited informative samples acquired via AL queries are more advantageous for early adaptation than CF samples. This does not last long, however, and OAL-CF takes the lead after session 10. This switch likely happens because the CF samples quickly reach the critical mass of adaptation samples required for the classifier to make significant improvements in performance.

Finally, the bottom panel of Fig. 7 shows the difference between the pre- and postquential trends. This difference trend specifically indicates how much the algorithm benefits from the AL queries over time. The closer to zero the curve gets, the more the classifier can rely on previously gathered annotations rather than the newest AL queries. Clearly, after the confusion of the first few samples, OAL-CF has a consistently smaller difference in pre- and postquential IMLM than OAL. This is expected, as CF gives the OAL-CF paradigm a much larger pool of annotations from previous sessions to work with compared to OAL. Understanding this difference is important for increasing the efficiency of the algorithm, as algorithms that do not rely heavily on AL queries have the option to reduce the budget, thereby reducing the IMLM.

Here, we can also compare the scores trend after the classifier has been updated with the new CF samples (i.e., the "post-CF" trend) to the pre- and postquential trends of the OAL-CF paradigm. Fig. 8 shows the prequential, postquential, and post-CF curves for the OAL-CF paradigm in terms of both IMLM and DCF. For reference, the overall post-CF IMLM score is 0.032 while the postquential score is 0.049, which is a significant difference (see Section VI-G for more details).

As expected, the added information about the current session from CF is very influential, causing the post-CF curve to fall well below even the postquential curve. While using the CF information in the same session from which it was acquired is not part of the OAL-CF paradigm as it was described in Section III-B, it does confirm that the information gained is being utilized effectively in the training process. It also indicates that repeated iteration of AL and CF steps on the one session can reduce error rate enough that the added effort required of the expert is worthwhile. We intend to explore this further in our future work.

D. Class Balance Analysis

It has been noted in the literature that class imbalance in the training set can be detrimental to performance, especially in online learning scenarios [24], [60], [61]. In Fig. 9, which shows the target class prevalence in the training data throughout the data stream, we see that OAL-CF consistently maintains close to 50% target samples in the adaptation pool while the other paradigms are consistently much lower. OAL-CF naturally maintains this balance because CF collects both false positives (i.e., non-target samples) and true positives (i.e., target samples). This is conceptually significantly different from other methods of achieving balanced classes, such as decimation or repetition of training samples. This feature of OAL-CF may be another reason for its low error rate.

E. Order Analysis

One question that is unique to online learning is the effect of the order of presentation on classifier performance. The order described in Section V-A1 was used for all baseline experiments, but here we also analyze the reverse order, as well as a uniformly distributed random order (e.g., the order shown in the lower panel

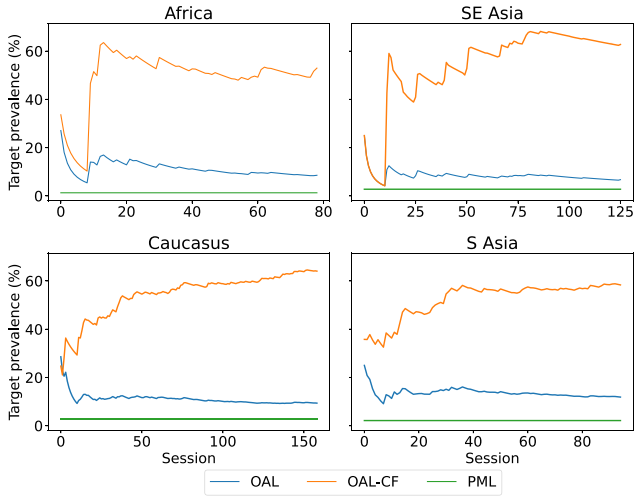


Fig. 9. Prevalence of the target class in the adaptation pool over time for all paradigms in each SLV corpus. OAL-CF achieves close to 50% target, 50% non-target for the majority of sessions. The other paradigms do not.

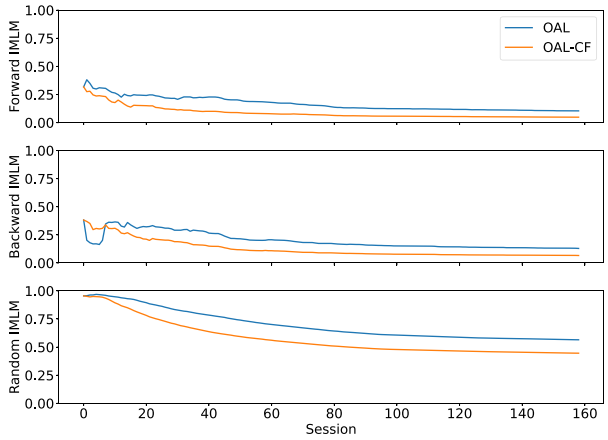


Fig. 10. The postquential IMLM trends for the OAL and OAL-CF paradigms applied to the SLV corpora run in normal, reverse, and random orders. Note that the normal and reverse orders exhibit similar behaviors while random ordering causes learning to slow down significantly.

of Fig. 4). The postquential trends for the three different orders are displayed in Fig. 10.

First, comparing the forward and backward plots to each other, we can see that order of presentation does have an effect on performance, even when samples from each class are temporally clustered in the same way. OAL performs slightly better than OAL-CF at the beginning of the backward plot, and the curves are located higher overall than in the forward plot. Despite this difference, however, we see that the forward and backward trends have approximately the same shape, starting with a higher IMLM, experiencing a few bumps within the early sessions, and then eventually smoothing out.

Comparing the forward and backward trends to the random order trend, we see a clear difference. Not only does the random plot start and stay rather high, it is also smooth for the whole duration of the runs. The overall poor performance can be attributed, at least in part, to how the uniform distribution of class samples eliminates the advantage of using context in the features—when there is no consistency in which samples are near each other, context is meaningless. The reason for the smooth shape is likely because every session has approximately the same number of target samples, causing the error rates and the state of the classifier to change consistently over time. The trends for the temporally clustered distributions are much more bumpy because some sessions have no target samples while others have many, so the change in FPR and FNR from session to session is inconsistent.

Thus, from this analysis, we can conclude that the OAL and OAL-CF algorithms described in this paper are better suited for data streams that contain more realistic class distributions with temporal clustering, as opposed to streams with uniform class distributions. We also conclude that, for any kind of class distribution, OAL-CF will outperform OAL on average.

TABLE VI
VTD PERFORMANCE RESULTS FOR THE THREE LEARNING PARADIGMS. OAL-CF YIELDS THE BEST PERFORMANCE OVERALL FOR EVERY METRIC.

Paradigm	IMLM	DCF	FNR	FPR
PML	15.569	0.096	0.101	0.083
OAL	0.073	0.039	0.046	0.019
OAL-CF	**0.054**	**0.029**	**0.037**	**0.007**

TABLE VII
SIGNIFICANCE TESTS COMPARING THE THREE PARADIGMS ACROSS VARIOUS EXPERIMENTS AND EVALUATION METRICS. THE TESTS INDICATE THAT OAL-CF IS BETTER THAN THE OTHER PARADIGMS WITH OVER 99% CONFIDENCE.

Task(s)	Para. 1	Para. 2	IMLM p-value	DCF p-value
SLV	OAL	PML	4.4×10^{-5}	7.3×10^{-2}
	OAL-CF	PML	4.4×10^{-5}	3.1×10^{-2}
	OAL-CF	OAL	1.8×10^{-3}	1.8×10^{-3}
VTD	OAL	PML	9.0×10^{-3}	1.2×10^{-1}
	OAL-CF	PML	9.0×10^{-3}	2.1×10^{-2}
	OAL-CF	OAL	9.0×10^{-3}	9.0×10^{-3}
SLV+VTD	OAL	PML	2.8×10^{-6}	1.9×10^{-2}
	OAL-CF	PML	2.8×10^{-6}	2.1×10^{-3}
	OAL-CF	OAL	1.6×10^{-4}	1.9×10^{-4}

F. Demonstration on VTD

In addition to SLV, we also compare the three paradigms on the VTD task. The results of these trials are shown in Table VI. Clearly, OAL-CF results in the best performance, and statistical tests in Section VI-G show that the improvement over other paradigms is significant once again. These results are important because they indicate that OAL-CF and the default parameter settings used in this paper generalize well to tasks other than SLV.

Comparing the online paradigms for each environment (see Fig. 11) reveals that OAL-CF performs better than OAL in every environment for the VTD task.

G. Significance Testing

To show the statistical significance of the improvement between experiments, we perform a single-tailed Wilcoxon

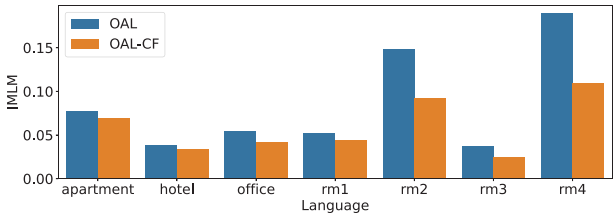


Fig. 11. Comparison of OAL and OAL-CF for each environment of the VTD corpora. OAL-CF yields lower IMLM for all 7 VTD environments.

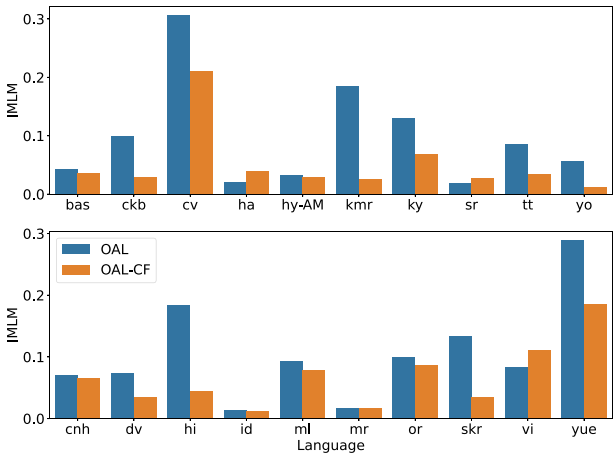


Fig. 12. Comparison of OAL and OAL-CF for each target language in the SLV corpora. OAL-CF yields lower IMLM for 17 of the 20 languages.

signed-rank test [62]. We compare the cumulative IMLM and DCF of each pair of runs across all languages. We report the p-values based on the hypothesis that paradigm 1 is significantly lower than paradigm 2 for the indicated metric.

Based on these p-values shown in Table VII, we can say with more than 90% confidence that OAL is better than PML, and with more than 99% confidence that OAL-CF is better than OAL and PML in terms of both IMLM and DCF for both tasks. The one exception is the comparison of OAL and PML considering only the VTD task, which is just outside of the 90% confidence range. This exception can be attributed to PML outperforming OAL in one of only seven environments.

Furthermore, we compare the post-CF results and the OAL-CF results in the same manner. For both IMLM and DCF, the Wilcoxon test indicates that post-CF scores are significantly better with a p-value of 2.2×10^{-4}. So, we can say with nearly 100% confidence that applying the information gleaned from CF to the current session yields an improvement compared to when CF is not applied.

VII. Limitations

While OAL-CF achieves statistically superior performance compared to OAL and PML in general, the paradigm does have limitations.

First, taking a closer view of the problem, Fig. 12 shows that OAL-CF achieves the best score for most, but not all, individual languages in the SLV corpora compared to OAL. (PML results were not included in this figure because they were too large and would dwarf the online results.) Specifically, OAL outperforms OAL-CF slightly for Hausa and Serbian, and more substantially for Vietnamese. The reasons for this particular behavior are unknown at this time, but they are indicative that the annotations obtained via CF occasionally do not result in improved performance. More efficient use of CF samples will be an area of future focus.

Second, OAL-CF continues to exhibit the problem of information saturation when selecting AL queries, as seen in many AL query selection strategies [1], [17]. This means that the query selection strategy becomes more and more random over time and the information gleaned from the queries diminishes. This is particularly dangerous in the case of OAL-CF, which depends on the AL channel to identify potential anomalies in the data stream so that the classifier can adapt quickly.

Third, while the addition of CF does allow the initial model to converge faster than OAL alone, the classifier remains relatively weak in the early sessions (see Figs. 8, 7, and 10).

Finally, only one round of CF acquisition is permitted per session in the OAL-CF paradigm as it is presented here. Alternatively, CF acquisition could be performed iteratively within each session to ensure that the classifier is as accurate as possible and that no target samples are missed. Limiting CF to a single iteration per session does restrict the effort required of the expert per session, but it may not give the paradigm the needed flexibility to perform at maximum efficiency. Given the promising post-CF scores from Section VI-C and the proven significant improvement they achieve from Section VI-G, we intend to explore iterative CF in the near future.

VIII. Conclusion

In this paper, we introduced a new online machine learning paradigm, Online Active Learning with Corrective Feedback (OAL-CF). This paradigm is an extension of the Online Active Learning (OAL) paradigm where Corrective Feedback (CF) from the domain expert controlling the operation is used as an additional information channel along with the Active Learning (AL) queries. The goal of the OAL-CF paradigm in this paper was to provide an efficient method for the domain expert to perform his or her assigned task and simultaneously train a classifier to aid in that task.

We showed experimentally that the OAL-CF paradigm is indeed an efficient method by running experiments using two streaming tasks—Spoken Language Verification and Voice-Type Discrimination. The performance of this paradigm was evaluated and compared to conventional Passive Machine Learning (PML) and OAL paradigms using metrics that account for the cost of both classification error rate and required annotation effort. The evaluations showed that OAL-CF resulted in a significant 53% decrease in cost compared to OAL.

Additional analysis showed that OAL-CF yields superior performance in nearly every individual trial, and at every stage of the training process. OAL-CF is also advantageous for varying orders of presentation of the data stream, AL budgets, and tasks.

It also maintains a relatively even balance between the classes in the pool of annotated data.

Future work that builds upon this paper may include improvements to specific pieces of the pipeline, such as AL query selection, CF sample usage, and adaptation methods. Such improvements should be aimed at providing some level of guarantee that the addition of CF will make OAL-CF perform better than OAL in all cases. Finally, the OAL-CF paradigm itself can be extended further by allowing repeated iterations of annotation and adaptation within each session of data. Such improvements and extensions can ensure that error rate is reduced to near zero, and that important target samples are all detected successfully.

ACKNOWLEDGMENT

Much of this work was completed as part of the first author's doctoral dissertation in the Department of Electrical and Computer Engineering at Carnegie Mellon University [1]. This work was made possible by a corporate gift from Probity Inc.

REFERENCES

[1] M. Lindsey, "Online active learning with corrective feedback: A new paradigm applied to streaming audio," Ph.D. dissertation, Dept. Elect. and Comput. Eng., Carnegie Mellon Univ., Pittsburgh, PA, USA, 2024.

[2] T. Brown et al., "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1877–1901.

[3] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," 2022, *arXiv:2204.06125*.

[4] C. Wang et al., "Neural codec language models are zero-shot text to speech synthesizers," 2023, *arXiv:2301.02111*.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[6] S. S. Paul, "Active and passive learning: A comparison," *GRD J. Eng.*, vol. 2, no. 9, pp. 27–29, 2017.

[7] D. Cacciarelli and M. Kulahci, "Active learning for data streams: A survey," *Mach. Learn.*, vol. 113, no. 1, pp. 185–239, 2024.

[8] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Proc. 17th Braz. Symp. Artif. Intell.*, Sao Luis, Maranhao, Brazil, Springer, 2004, pp. 286–295.

[9] D. Sculley, "Online active learning methods for fast label-efficient spam filtering," in *Proc. Conf. Email Anti-Spam*, 2007, pp. 143–150.

[10] J. Kranjc, J. Smailović, V. Podpečan, M. Grčar, M. Žnidaršič, and N. Lavrač, "Active learning for sentiment analysis on data streams: Methodology and workflow implementation in the ClowdFlows platform," *Inf. Process. Manage.*, vol. 51, no. 2, pp. 187–203, 2015.

[11] A. Narr, R. Triebel, and D. Cremers, "Stream-based active learning for efficient and adaptive classification of 3D objects," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 227–233.

[12] D. Manjah et al., "Stream-based active distillation for scalable model deployment," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 4999–5007.

[13] S. Ghiasi, G. Pazzi, C. D. Grosso, G. D. Magistris, and G. Veneri, "Combining thermodynamics-based model of the centrifugal compressors and active machine learning for enhanced industrial design optimization," 2023. [Online]. Available: https://arxiv.org/abs/2309.02818

[14] X. Yan et al., "An online learning framework for sensor fault diagnosis analysis in autonomous cars," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 14467–14479, Dec. 2023.

[15] R. Schmitt, P. Jatzkowski, and M. Peterek, "Traceable measurements using machine tools," in *Proc. 10th Int. Conf. Exhib. Laser Metrol., Mach. Tool, CMM Robotic Perform.*, Lamdamap, 2013, pp. 20–21.

[16] N. Beck, S. Kothawade, P. Shenoy, and R. Iyer, "STREAMLINE: Streaming active learning for realistic multi-distributional settings," 2023. [Online]. Available: https://arxiv.org/abs/2305.10643

[17] M. Lindsey, N. R. Robinson, F. Kubala, and R. M. Stern, "Reducing the cost of spoof detection labeling using mixed-strategy active learning and pretrained models," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2023, pp. 1–7.

[18] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 12, pp. 2346–2363, Dec. 2019.

[19] G. J. Aguiar and A. Cano, "Dynamic budget allocation for sparsely labeled drifting data streams," *Inf. Sci.*, vol. 654, 2024, Art. no. 119821.

[20] Z. Wu, H. Wang, J. Guo, Q. Yang, and J. Shao, "Learning evolving prototypes for imbalanced data stream classification with limited labels," *Inf. Sci.*, vol. 679, 2024, Art. no. 120979.

[21] B. Krawczyk, B. Pfahringer, and M. Woźniak, "Combining active learning with concept drift detection for data stream mining," in *Proc. IEEE Int. Conf. Big Data*, 2018, pp. 2239–2244.

[22] W. Xu, F. Zhao, and Z. Lu, "Active learning over evolving data streams using paired ensemble framework," in *Proc. 8th Int. Conf. Adv. Comput. Intell.*, 2016, pp. 180–185.

[23] Y. Chen and H. Jin, "Rare sound event detection using deep learning and data augmentation," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 619–623.

[24] Ł. Korycki and B. Krawczyk, "Online oversampling for sparsely labeled imbalanced and non-stationary data streams," in *Proc. Int. Joint Conf. Neural Netw.*, 2020, pp. 1–8.

[25] Ł. Korycki and B. Krawczyk, "Low-dimensional representation learning from imbalanced data streams," in *Proc. Pacific-Asia Conf. Knowl. Discov. Data Mining*, Springer, 2021, pp. 629–641.

[26] M. Lindsey, A. Shah, F. Kubala, and R. M. Stern, "Online active learning for sound event detection," 2023. [Online]. Available: https://arxiv.org/abs/2309.14460

[27] S. Hao, P. Hu, P. Zhao, S. C. Hoi, and C. Miao, "Online active learning with expert advice," *ACM Trans. Knowl. Discov. Data*, vol. 12, no. 5, pp. 1–22, 2018.

[28] P. Donmez and J. G. Carbonell, "Proactive learning: Cost-sensitive active learning with multiple imperfect oracles," in *Proc. 17th ACM Conf. Inf. Knowl. Manage.*, 2008, pp. 619–628.

[29] A. Castellani, S. Schmitt, and B. Hammer, "Stream-based active learning with verification latency in non-stationary environments," in *Proc. Int. Conf. Artif. Neural Netw.*, Springer, 2022, pp. 260–272.

[30] E. Mosqueira-Rey, E. Hernandez-Pereira, D. Alonso-Rios, J. Bobes-Bascaran, and A. Fernandez-Leal, "Human-in-the-loop machine learning: A state of the art," *Artif. Intell. Rev.*, vol. 56, no. 4, pp. 3005–3054, 2023.

[31] B. Settles, "Active learning literature survey," Dept. Comput. Sci., Univ. Wisconsin-Madison, Tech. Rep., 2009.

[32] J. J. Dudley and P. O. Kristensson, "A review of user interface design for interactive machine learning," *ACM Trans. Interactive Intell. Syst.*, vol. 8, no. 2, pp. 1–37, 2018.

[33] M. Ware, E. Frank, G. Holmes, M. Hall, and I. H. Witten, "Interactive machine learning: Letting users build classifiers," *Int. J. Hum.- Comput. Stud.*, vol. 55, no. 3, pp. 281–292, 2001.

[34] J. Bernard, M. Hutter, M. Zeppelzauer, D. Fellner, and M. Sedlmair, "Comparing visual-interactive labeling with active learning: An experimental study," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 298–308, Jan. 2018.

[35] T. Ishibashi, Y. Nakao, and Y. Sugano, "Investigating audio data visualization for interactive sound recognition," in *Proc. 25th Int. Conf. Intell. User Interfaces*, 2020, pp. 67–77.

[36] L. Begeja, B. Renger, D. Gibbon, Z. Liu, and B. Shahraray, "Interactive machine learning techniques for improving SLU models," in *Proc. HLT-NAACL 2004 Workshop Spoken Lang. Understanding Conversational Syst. Higher Level Linguistic Inf. Speech Process.*, 2004, pp. 10–16.

[37] J. Talbot, B. Lee, A. Kapoor, and D. S. Tan, "EnsembleMatrix: Interactive visualization to support machine learning with multiple classifiers," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 2009, pp. 1283–1292.

[38] J. A. Fails and D. R. Olsen Jr., "Interactive machine learning," in *Proc. 8th Int. Conf. Intell. User Interfaces*, 2003, pp. 39–45.

[39] A. Kumar, A. Gupta, and S. Levine, "DisCor: Corrective feedback in reinforcement learning via distribution correction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 18560–18572.

[40] C. Celemin, J. Ruiz-del Solar, and J. Kober, "A fast hybrid reinforcement learning framework with human corrective feedback," *Auton. Robots*, vol. 43, pp. 1173–1186, 2019.

[41] R. Pérez-Dattari, C. Celemin, J. Ruiz-del Solar, and J. Kober, "Interactive learning with corrective feedback for policies based on deep neural networks," in *Proc. Int. Symp. Exp. Robot.*, Springer, 2020, pp. 353–363.

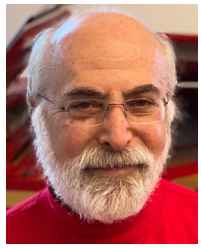
[42] A. Y. Ng et al., "Algorithms for inverse reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2000, pp. 663–670.

[43] L. Shani, T. Zahavy, and S. Mannor, "Online apprenticeship learning," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 8240–8248.

[44] M. Lindsey, F. Kubala, and R. M. Stern, "A unified metric for simultaneous evaluation of error rate and annotation cost," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2025, pp. 1–5.
[45] A. Martin and M. Przybocki, "The NIST 1999 speaker recognition evaluation–An overview," *Digit. Signal Process.*, vol. 10, no. 1, pp. 1–18, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S105120049990355X
[46] F. Byers, J. Fiscus, G. SeyedSanders, and M. Przybocki, "Open speech analytic technologies pilot evaluation OpenSAT pilot," 2019. [Online]. Available: https://www.nist.gov/sites/default/files/documents/2018/07/05/nist2017pilotopensatevalplanfinal.pdf
[47] J. Gama, R. Sebastiao, and P. P. Rodrigues, "Issues in evaluation of stream learning algorithms," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 329–338.
[48] M. Khannouz and T. Glatard, "Dynamic ensemble size adjustment for memory constrained Mondrian forest," in *Proc. IEEE Int. Conf. Big Data*, 2022, pp. 3358–3363.
[49] C. M., A. Mandal, and S. Mukherjee, "Study of ECAPA-TDNN models for spoken language identification task," in *Proc. IEEE World Conf. Appl. Intell. Comput.*, 2023, pp. 233–237.
[50] J. Yamagishi et al., "ASVspoof 2019: Automatic speaker verification spoofing and countermeasures challenge evaluation plan," *ASV Spoof*, Tech. Rep., 2019.
[51] R. Ardila et al., "Common voice: A massively-multilingual speech corpus," 2019, *arXiv: 1912.06670*.
[52] L. Yuan, B. Xie, and S. Li, "Robust test-time adaptation in dynamic scenarios," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 15922–15932.
[53] M. Lindsey, T. Vuong, and R. M. Stern, "Unsupervised voice type discrimination score adaptation using X-vector clusters," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2023, pp. 1–5.
[54] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification," 2020, *arXiv: 2005.07143*.
[55] J. Valk and T. Alumäe, "VoxLingua107: A dataset for spoken language recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 652–658.
[56] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 5329–5333.
[57] S. Chen et al., "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 6, pp. 1505–1518, Oct. 2022.
[58] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 1735–1742.
[59] G. King and L. Zeng, "Logistic regression in rare events data," *Political Anal.*, vol. 9, no. 2, pp. 137–163, 2001.
[60] S. Ertekin, J. Huang, and C. L. Giles, "Active learning for class imbalance problem," in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2007, pp. 823–824.
[61] S. Ertekin, J. Huang, L. Bottou, and L. Giles, "Learning on the border: Active learning in imbalanced data classification," in *Proc. 16th ACM Conf. Inf. Knowl. Manage.*, 2007, pp. 127–136.
[62] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, 1945.

Mark Lindsey (Member, IEEE) received the BS degree in electrical engineering from Brigham Young University, Provo, Utah, in 2019. He completed his doctoral thesis, "Online Active Learning with Corrective Feedback: A New Paradigm Applied to Streaming Audio", in 2024 with Carnegie Mellon University, Pittsburgh, Pennsylvania, under advisor Dr. Richard Stern with support from Probity Inc., after which he joined Probity's Content Analytics Division as a Research and Development Engineer in Herndon, Virginia. His research interests include human-in-the-loop algorithms, online learning, machine learning, computational linguistics, and audio signal processing. His research has been published with venues, such as ICASSP and ASRU.

Francis Kubala (Member, IEEE) received the bachelor of science degree in electrical engineering from the Massachusetts Institute of Technology, in 1984. He joined BBN Technologies in Cambridge, Massachusetts just as the Defense Advanced Research Projects Agency (DARPA) began ramping up its investment in large vocabulary, speaker independent Automatic Speech Recognition (ASR), which continued for the next 20 years. He was a seminal contributor to the standardized international technology evaluations administered by the US National Institute of Standards and Technology (NIST) beginning, in 1986 and beyond. He was a performer or Principal Investigator in every DARPA-sponsored ASR and speech-related program through 2007. His focus was multilingual triage from streaming media. His team developed the first real-time rich-transcription engine for English and the first real- time satellite news transcription and translation system for non-English media that was deployed worldwide. His research interests remain focused on adaptive algorithms that improve by leveraging feedback from non-technical users while operationally deployed.



Richard M. Stern (Fellow, IEEE) received the BS degree from the Massachusetts Institute of Technology, in 1970, the MS degree from the University of California, Berkeley, in 1972, and the PhD degree from MIT, in 1977, all in electrical engineering. He has been on the faculty of Carnegie Mellon University (CMU) since 1977, where he is currently a professor with the Electrical and Computer Engineering Department, the Computer Science Department, and the Language Technologies Institute. He is also an artist lecturer in CMU's School of Music. Much of his research has been in spoken language systems, where he was particularly concerned with the development of techniques with which automatic speech recognition can be made more robust with respect to changes in environment and acoustical ambience. He has also developed sentence parsing and speaker adaptation algorithms for earlier CMU speech systems. In addition to his work in speech recognition, he also maintained an active research program in psychoacoustics, where he is best known for theoretical work in binaural perception. He is a fellow of the Acoustical Society of America, and the International Speech Communication Association (ISCA), and he was a recipient of the Allen Newell Award for Research Excellence, in 1992. He served as the general chair of Interspeech 2006 and as the 2008–2009 ISCA distinguished lecturer. He is also a member of the Audio Engineering Society.