The Interactive Machine Learning Paradigm

Submitted as a prospectus for

the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Mark Lindsey

B.S., Electrical Engineering, Brigham Young University

Graduate Supervisory Committee:

Dr. Richard M. Stern, Chair
Dr. Bhiksha Raj
Dr. Aswin Sankaranarayanan
Dr. Rita Singh
Francis Kubala

Carnegie Mellon University
Pittsburgh, PA

June 2023

Abstract

Machine Learning (ML) is a powerful tool that can assist humans in doing a wide variety of tasks. However, typical ML paradigms tend to remove the human element from the equation as frequently as possible in the name of automation. This approach may be useful for creating powerful solutions to a selection of pre-defined tasks, but it has lead to a dearth of methods for quickly training a machine to assist a human domain expert in a task previously undefined for ML. This problem is compounded in cases where data for the intended task are scarce or proprietary, since typical ML paradigms rely on large amounts of training data. To address this and other related issues with typical ML paradigms, I propose an Interactive Machine Learning (IML) paradigm in which a machine classifier is trained to perform a new task alongside a human domain expert as part of his or her operational workflow.

This proposal includes a description of the IML paradigm and how it differs from other existing paradigms, as well as preliminary results showing the efficacy of the IML paradigm compared to typical supervised training. It also includes three pieces of proposed work that build on the existing IML pipeline to jointly optimize the performance of the classifier being trained and the amount of feedback required of the domain expert. These include a mathematical formulation of the IML objective function based on the joint optimization problem, a method for handling weighted error functions and imbalanced class distributions, and methods for exploiting the temporal nature of the IML training process.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The common approach adopted by many modern Machine Learning (ML) training paradigms is to train classifiers on massive amounts of data with no intervention from the end user. The advantages of training in this fashion are readily apparent, since reliable, off-the-shelf solutions to problems are much more convenient to a user than something that must be configured or calibrated.

Despite this convenience, however, this type of solution is not optimal for every situation. Applications where a human expert is working alongside an ML classifier to perform a highly specialized, error-sensitive task are a prime example of situations where the typical training paradigm is clearly disadvantaged. Classifiers trained on a massive amount of widely ranging data fall short for this type of application in the following ways. First, these classifiers are not specialized, as they often sacrifice precision in a limited problem space for reasonable performance in the larger problem space, which can be a significant risk for sensitive tasks. Second, off-the-shelf or pre-trained solutions do not allow for continual adjustment. This can be a serious issue, as unseen or unexpected data will inevitably exploit weaknesses in the pre-trained classifier if it is unable to adapt. Furthermore, the task itself may not be well-defined before the expert begins working on it, requiring continued tuning and tweaking until it is defined. Finally, removing user input from the pipeline can be detrimental to classification performance because the user, in this case, is a domain expert who defines the task and can provide crucial operational information about the data.

To address these issues, I propose an Interactive Machine Learning (IML) paradigm (see Fig. 1.1) which trains and continually adapts a classifier to perform a custom, specialized task given frequent feedback from a human domain expert working on the same task. This proposal is intended as a solution for binary audio detection tasks, but it will lay the groundwork for application to any detection task. In this paradigm, a seed classifier starts with a very limited ability to perform the task, then uses a series of

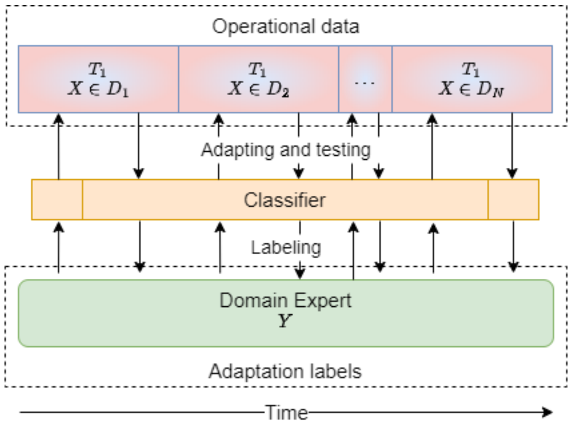

Figure 1.1: A high-level view of the IML paradigm. Unlabeled data (X) from potentially different distributions (D_n) for a single task (T_1) are evaluated over time by the classifier. Some adaptation labels (Y) for the operational data are provided by the domain expert. This way, the classifier is trained on a labeled subset of the operational data, and then used to classify the rest.

periodic, limited interactions with the domain expert to shape itself incrementally into a tool that can effectively aid the domain expert in his or her work. Interactions with the domain expert include initializing the classifier on a small bootstrap dataset prepared by the expert for the operating environment, querying the expert about samples that might be informative (i.e., Active Learning (AL)), and collecting corrective feedback from the expert about the classifier's predictions (i.e., Corrective Feedback (CF)). The goal of the paradigm is to optimize the tradeoff between classification performance (i.e., how much the classifier reduces the domain expert's workload) and the amount of human intervention required (i.e., how much the training procedure increases the domain expert's workload).

The IML paradigm is expected to be successful for two primary reasons. First and foremost, the data used to train and adapt the classifier in the IML paradigm are sampled by the domain expert directly from the evaluation data, which nearly eliminates any possibility of domain mismatch. Secondly, the paradigm leverages pretrained embeddings, which provide a reliable representation on which a small classifier can be initialized and adapted to perform each task quickly and precisely [1].

To guarantee the success of the IML paradigm, however, the question of how to optimize the tradeoff between classifier performance and the amount of required human intervention must be answered. This is a question unique to the IML paradigm and the primary research question my proposed thesis will address. As such, work for the thesis will include contributions in several related areas, including the characterization of the tradeoff with a mathematical classifier, methods for choosing AL queries to balance false negative and false positive errors, and methods for adjusting AL queries over time in the IML pipeline to optimize the tradeoff.

The remainder of this document is organized as follows. Chapter 2 clarifies the difference between IML and other similar ML paradigms. Chapter 3 describes the details of the IML paradigm in depth. Chapter 4 outlines the proposed work that will contribute to the thesis. Finally, Chapter 5 shows the timeline in which the work will be completed.

Chapter 2

Related Work

The IML paradigm is similar to some existing ML training paradigms in many ways, but ultimately unique. This chapter contrasts the IML paradigm with the three most similar existing paradigms, including Continual Learning (CL), Domain Adaptation (DA), and Active Learning (AL).

2.1 Continual Learning

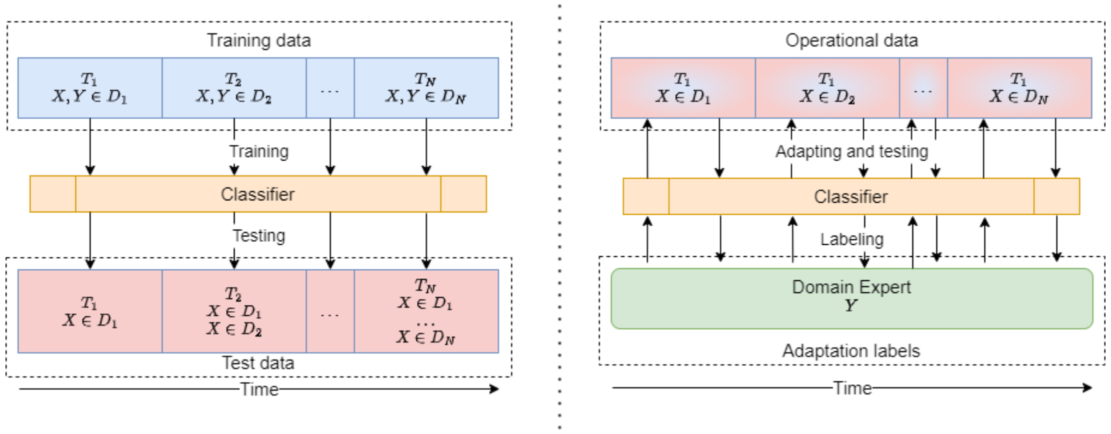

Figure 2.1: Contrasting CL (left) and IML (right). CL learns many tasks (T_n) over time and separates unlabeled test data from labeled training data. IML only learns one task (T_1) and obtains labels (Y) for a subset of the training data from the domain expert.

CL, which is also frequently referred to as Lifelong Learning and Incremental Learning, is defined in [2] as “an adaptive algorithm capable of learning from a continuous stream of information, with such information becoming progressively available over time and where the number of tasks to be learned (e.g., membership classes in a classification task) are not predefined. Critically, the accommodation of new information should occur without catastrophic forgetting or interference.” Here, catastrophic forgetting refers to when performance on an old task drastically deteriorates when a new one is learned. Interference is any significant decrease in performance on old tasks, whether large or small.

IML and CL share a similar conceptual goal, which is to allow a classifier to accrue knowledge and ability over time. However, the two paradigms differ significantly in their definitions of knowledge and ability, and, thus, how these traits are measured. Specifically, CL typically measures how well the classifier has learned to perform a set of tasks over time, while IML tracks the classifier's performance on a single task over time. Additionally, CL is concerned with catastrophic forgetting. IML, on the other hand, can still be considered successful if it forgets past knowledge, so long as forgetting does not affect future performance.

It is also important to note that CL is not defined to include a human interaction component, but human interaction is an integral part of IML, as shown in Fig. 2.1. IML also does not have a held-out test set, since a subset of the operational data is used for supervised adaptation.

Despite the clear differences between CL and IML, some CL concepts may prove effective when applied to the IML paradigm. For example, CL methods that address catastrophic forgetting by keeping the most informative training samples in memory [3, 4, 5] could be useful in selecting samples to be labeled for the IML adaptation cycle.

2.2 Domain Adaptation

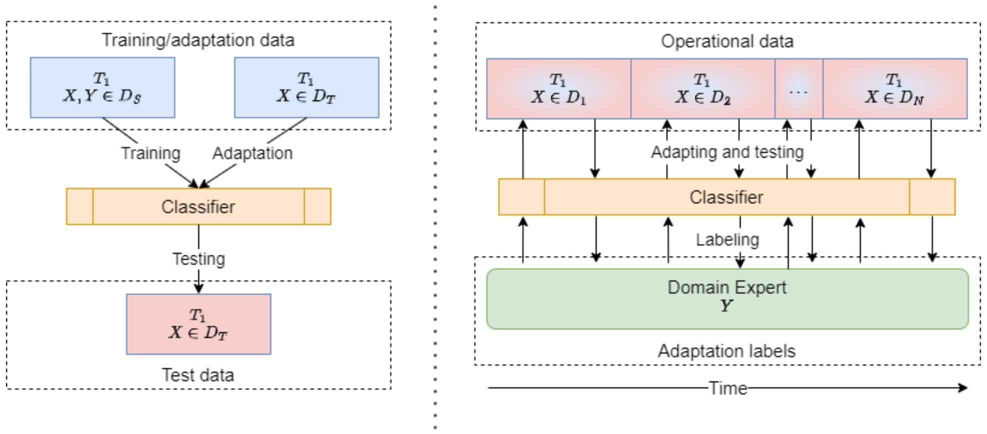

Figure 2.2: Contrasting DA (left) and IML (right). DA usually transfers the classifier from the source distribution (D_S) to the target distribution (D_T) in one step and does not obtain labels from a domain expert.

DA is the process of adapting a classifier originally trained on a task on data from a source distribution to perform the same task on data from a separate target distribution while mitigating any performance degradation caused by domain mismatch.

IML differs significantly from DA in that IML is an iterative process that occurs while the classifier is being trained and used, while DA is generally a one-time process that occurs after the initial training on the source distribution but before deployment of the classifier. The only significant similarity between IML and DA is the concept of adapting to a new distribution which was previously unseen by the classifier.

It should be noted, however, that IML is quite similar to a related subfield of DA called Continuous Domain Adaptation (CDA) [6, 7, 8]. In CDA, the classifier is adapted incrementally to handle a continuously evolving data distribution. This type of incremental adaptation is a key piece of IML. Additionally, CDA algorithms often include methods to automatically track potential drifts in the data distribution, a feature which might greatly benefit IML. However, CDA still lacks the human interaction component of IML, as many existing CDA methods are completely unsupervised.

2.3 Active Learning

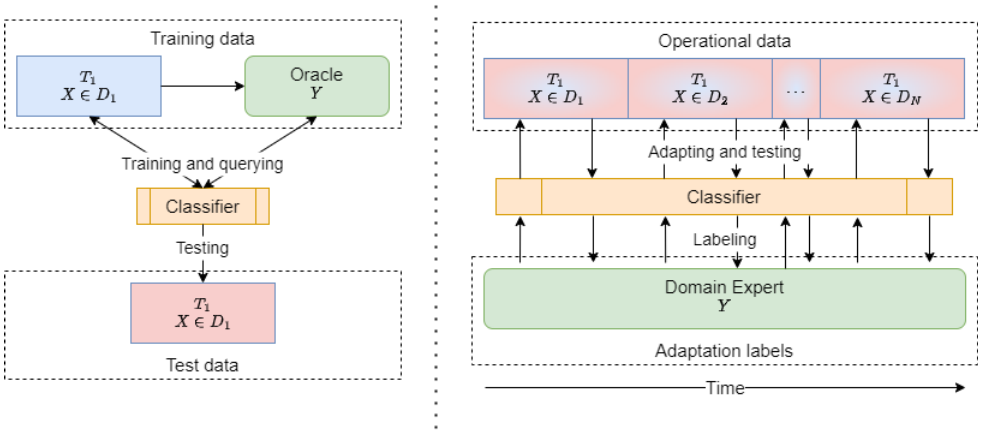

Figure 2.3: Contrasting AL (left) and IML (right). AL queries an oracle for a limited set of labels (Y) to optimize performance on fixed held-out test data for a single task (T_1). IML uses AL as one method of obtaining adaptation labels.

AL is an ML paradigm where an algorithm forms an optimal set of labeled training data from a larger set of unlabeled training data by querying an oracle (e.g., human annotator) about a limited number of labels. The goal of AL is to produce a classifier with a high level of accuracy using a small number of queries [9, 10].

A survey of AL methods refers to the approach to sampling unlabeled data points about which to query the oracle as the “query strategy" [9]. In this proposal, I refer to this algorithm as the *AL sampling method*. According to the survey, there are a number of classes of AL sampling methods, including (but not limited to) uncertainty sampling, expected model change, variance reduction, and estimated error reduction. Uncertainty sampling methods [11, 12, 13, 14], which are the most common method for AL sampling, attempt to identify the samples which are most likely to be classified incorrectly by the classifier, as measured by some objective metric. Expected model change methods [15] choose the samples that would change the model parameters the most if the labels for those samples were known. Variance reduction methods [16, 17, 18] leverage statistics to choose the samples that are predicted to minimize the variance of the classifier. Finally, estimated error reduction methods [19, 20, 21] work the same way as variance reduction methods, but the objective is to minimize the putative classification error rather than variance.

AL methods are used extensively in the IML paradigm as one of the channels for interacting with the domain expert. Note that IML contains AL in addition to other components (e.g., CF), so IML and AL are indeed different concepts. Also note that IML operates on a continually evolving set of data that changes as the domain expert finishes analysis and moves on. The assumption that the data are continually changing is not typically present in AL, and training progress is usually tracked using a held-out test set.

Chapter 3

Formulating Interactive Machine Learning

In contrast to existing Machine Learning (ML) training paradigms, I propose an Interactive Machine Learning (IML) paradigm for binary audio detection tasks. As stated in the introduction, IML is intended to act as a framework to address the lack of human interaction, environmental specialization, and adaptability in existing ML training paradigms. The IML paradigm serves as a method of training ML classifiers to aid a human operator in the task he or she is performing, with the objective of jointly minimizing classification error rate and the amount of feedback required of the operator.

This chapter describes the IML paradigm in detail.

3.1 Overview of the IML Paradigm

The IML paradigm is designed to train a classifier to perform a detection task as part of the operational workflow of a human with expertise in the task. That human is denoted here as the *domain expert*. The training paradigm allows the domain expert to define a task and teach a classifier to do that task precisely enough to yield a significant reduction in his or her workload.

The domain expert interacts with a classifier and teaches it to perform a task by periodically providing ground truth labels for the operational data in the specific *operating environment* where he or she is working. The operating environment is defined as a sensor in a constant location recording events that may or may not be changing. In an audio detection task, for example, an operating environment would consist of a microphone placed in a fixed location in an auditory scene with various auditory events happening around it.

The labeled data provided by the domain expert are added to a growing pool of adaptation data used to fine-tune the classifier during each *session*. A session is simply a predetermined length of audio presented at one time. For each session, the domain expert provides two forms of feedback: Active Learning (AL)

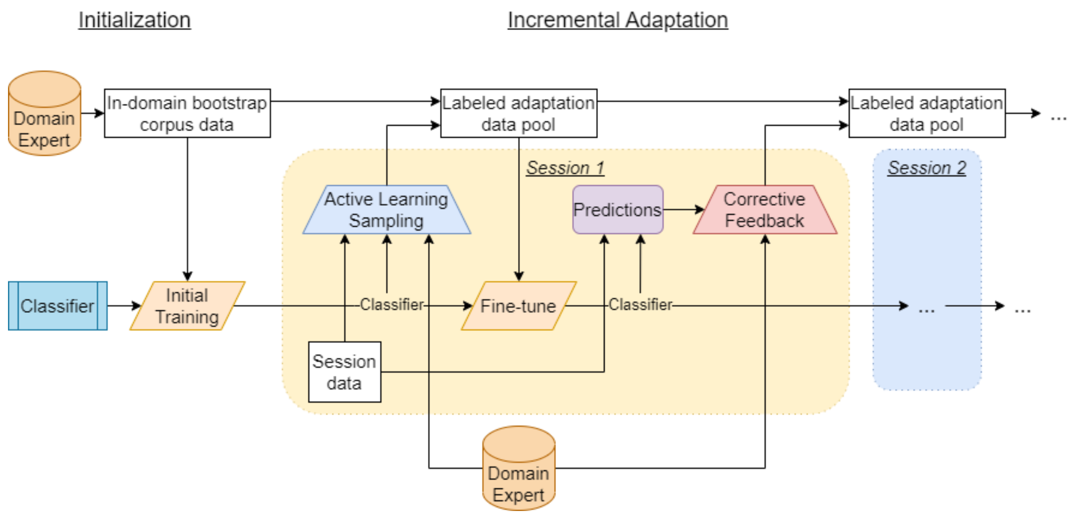

Figure 3.1: Overview of the IML paradigm.

feedback and Corrective Feedback (CF). AL feedback obtained as the domain expert responds to queries made by the system about the current session. CF is provided by the domain expert after he or she has seen the predictions. In operationally realistic conditions, the domain expert is expected to only review samples predicted to be from the target class, so CF will only be provided for these samples. This is an opportunity for the domain expert to correct repeating false positive errors. Since the focus of CF is the non-target samples, AL is used to acquire labels for target samples.

At the beginning of operation in a new environment, the domain expert also provides a *bootstrap corpus* of data for the purpose of initializing a classifier. The bootstrap corpus is characterized as being equally representative of both classes of the binary detection task and serves as the seed for the adaptation data pool.

The following subsections describe in detail the specifics of the IML paradigm and how it is used to train classifiers for different tasks.

3.2 Specification of the IML Paradigm

Fig. 3.1 illustrates how each component of the IML paradigm fits together, the distinction between the initialization phase and the adaptation phase, and the procedure for extracting task-related information from the domain expert. This section provides a detailed explanation of each of these components.

The first step in the process is initializing the classifier to a point where it can be used to make predictions in the operating environment that are not completely random. This is achieved by training a seed classifier on the bootstrap corpus. Since the bootstrap corpus is taken directly from the first sessions of the operating environment, the seed classifier is expected to perform reasonably well on the early sessions.

After the seed classifier has completed the initialization phase, it enters the incremental adaptation phase. Incremental adaptation consists of performing a set series of steps for each session in the operating environment. The first step is to add to the pool of labeled adaptation data by posing a limited number of AL queries to the domain expert about the ground truth labels for data from the current session. The subset of samples to be presented for queries is determined by an algorithm that is part of the IML pipeline. Methods for efficient AL sampling are explored in detail in Section 3.7.

After AL sampling, the classifier is fine-tuned using labeled data from the adaptation data pool. The classifier can be fine-tuned with all available data in the pool or selected data points from the pool.

The adapted classifier is then used to generate predictions for the current session. These predictions may optionally be adjusted using unsupervised learning or posterior error correction methods. The classifier's predictions for the current session are then presented to the domain expert for analysis.

In the course of their analysis, the domain expert will provide CF to the classifier, which serves as another contribution to the pool of labeled adaptation data. CF may come in the form of explicit CF from the domain expert as he or she attempts to correct repeated mistakes made by the classifier. It may also come in the form of implicit CF based on how the domain expert interacts with the data on the user interface. For example, the domain expert may skip a section of audio, indicating that it contains no important information for the task, and therefore no target samples. Since the detection tasks are constrained to be binary tasks, any indication of the class a sample does not belong to is just as good as an indication of the true class.

3.3 Simulating Human Feedback

For the purposes of developing and testing the IML paradigm, it is far more convenient to simulate the feedback a domain expert might provide than to have a domain expert present for every experiment. Formulation of the bootstrap corpus, AL sampling, and CF can be reasonably simulated using fully annotated development corpora.

The bootstrap corpus is intended to come from annotations made by the domain expert at the beginning of the operational environment. It is also intended to contain representative samples from both the target and non-target classes. Thus, a simulated bootstrap corpus can be created by looking up and storing the labels of the first N samples in the operating environment for each class, where N is a parameter that can vary based on the task or the experiment being performed.

For AL sampling, a response to a query can be simulated simply by looking up the corresponding ground truth label of the sample in question. For each session, M AL samples will be labeled and added to the adaptation data pool in this way, where M is a parameter that can vary depending on the task.

Implicit CF may take different forms in practice, depending on the task, the domain expert, and the interface between the machine and the user. However, explicit CF can be simulated somewhat realistically for a session by correcting sufficiently large regions of incorrect predictions in temporal order up to a specified limit. Specifically, the first C clusters of L consecutive incorrect predictions will be corrected and added to the adaptation data pool with the corresponding ground truth labels. If fewer than C clusters of L incorrect predictions exist, then all such clusters are corrected. Again, C and L are parameters that can be adjusted based on the task.

3.4 Application to Audio Detection Tasks

To show that the IML paradigm is practically useful, I intend to apply it to three different tasks: Voice-type Discrimination (VTD), Sound Event Detection (SED), and Spoof Detection (SD). This section describes each of these tasks and how they are adjusted to fit into the IML paradigm, as well as how a task trained in the paradigm is evaluated.

3.4.1 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 by a person while he or she is in physical proximity to the recording device. This means that speech being played through a loudspeaker into the recording device is not considered live speech, whether that speech was pre-recorded or broadcast from another location [22].

For VTD, the target class is composed of all samples containing live speech. Any sample that does not contain live speech is considered to be part of the non-target class. Samples are pre-defined as abutting segments of audio, five seconds in length. An operating environment for the VTD task is a single microphone array location in a specific room. A session is defined to be one hour long, which means there are 720 samples per session.

Data for the VTD task come from two corpora: the SRI Corpus and the Lionbridge (LB) Corpus. The SRI Corpus is composed of recordings made in four rooms, recorded from five microphone array locations in each room, resulting in 20 operating environments. The LB Corpus comes from recordings made in three rooms from seven microphones 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 the English, Mandarin Chinese, Spanish, and Russian languages.

In the VTD task, it is much more detrimental to the operation to miss target speech than it is to misclassify non-target audio. As such, the metric used for evaluation is the Detection Cost Function (DCF) shown in the equation below, where the False Negative Rate (FNR) is weighted three times as much as the False Positive Rate (FPR). To calculate an aggregated DCF over all sessions in a VTD operating environment, the number of false positive and false negative errors and true positive and true negative samples are accumulated after predictions are made for each session. These four numbers are used to calculate FNR, FPR, and then DCF.

3.4.2 Sound Event Detection

SED is the task of identifying the temporal location (i.e., onset and offset time) of a sound event in an audio recording [23]. A sound event can take the form of any type of sound, like a dog barking, water running, or a gun firing. In some contexts, SED is viewed as a multi-class problem where both the type of sound event and the temporal location of the sound event are to be identified. Here, SED is limited to the binary classification problem of identifying whether a pre-specified type of sound event occurred in a given audio region. Note that SED where the target sound event is live speech is essentially the same task as VTD.

Operating environment, session, and target classes for SED are defined very similarly to VTD. The target class consists of all samples containing the specified sound event; the non-target class is all the samples that do not contain that type of sound event. The operating environment is a microphone array in a fixed location within a single auditory scene. SED sessions are defined as sets of ten samples in a fixed temporal order, where one sample is ten seconds of contiguous audio.

SED data comes from the SONYC Urban Sound Tagging (SONYC-UST) dataset [24], which is composed of ten-second recordings of urban audio scenes by 60 different sensors placed throughout New York City. SONYC-UST contains 51 hours of audio total (185,150 ten-second samples) spread across the 60 sensors (i.e., operating environments). This means each operating environment is composed of approximately 30 sessions on average. The dataset comes with annotations indicating the presence of eight coarse classes of audio, any of which can be used as the target class.

The SED task is evaluated in the same way as VTD, using DCF aggregated over all sessions.

3.4.3 Spoof Detection

Voice spoofing is the act of tricking a human listener or an Automatic Speaker Verification (ASV) system by producing speech audio intended to sound identical to the speech of a specific speaker. SD is the task of identifying whether a voice recording is bona fide speech produced by the intended speaker or spoofed speech.

The ASVspoof 2019 physical attacks (PA) dataset [25] is used for the SD task. This dataset contains bona fide samples from 107 different speakers and spoofed samples in the form of voice recordings from the target speaker played back through a loudspeaker. Bona fide and spoofed samples are recorded in 27 different acoustic conditions, meaning that the dataset can be separated into 27 distinct operating environments. There are approximately 200,000 utterances in the PA dataset. A session for this task consists of 70 samples, resulting in 100 sessions per environment.

The evaluation metric for this task is the tandem Decision Cost Function (t-DCF) metric defined by the ASVspoof challenge [26]. This metric adjusts the typical DCF function to fit the SD task by assigning a higher weight to false positives than false negatives. t-DCF also combines the concepts of anti-spoofing countermeasures (i.e., SD) and ASV into one metric.

3.5 Discussion on Architecture and Features

The classifier used in the IML paradigm should be lightweight, easy to adapt, and capable of achieving a very low error in the operating environment given sufficient adaptation data. As such, I propose applying the same simple neural network architecture to each task. This architecture, shown in Fig. 3.2, is composed of a contrastive network and a classifier network. The contrastive network performs transfer learning on the input features to project the representation into two distinct clusters using a contrastive loss [27]. The rest of the network then classifies the projected features into the target and non-target classes.

A strong input representation is crucial for the success of such a simple classifier. It has already been shown that x-vectors [28] work quite well for the VTD task. Since SED is not necessarily a speech-related task, the generic audio representation CLAP [29] will be used as the input feature. SD, like VTD, is a speech task, but it may benefit from information other than just speaker identity. Thus, WavLM [30] embeddings will be used as the SD input feature.

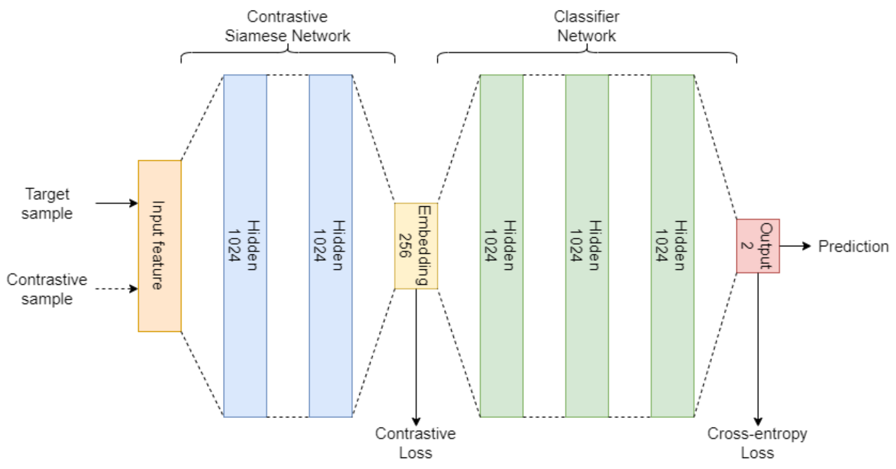

Figure 3.2: The contrastive classifier architecture

3.6 Benchmarking

Typical ML solutions to each of the tasks outlined in Section 3.4 already exist. To substantiate the claim that the IML paradigm is advantageous compared to these solutions, it is important to make a fair, objective comparison in terms of performance and the amount of labeled data used for training or adaptation. Since the IML paradigm uses a subset of the operational data for initialization and adaptation, it could be said that IML is “testing on the training data". To handle this issue, all samples used for adaptation are not counted during evaluation. That same set of data with adaptation samples removed is used to evaluate both the baseline classifier and the IML-trained classifier.

In the next section, preliminary IML results for the VTD task are presented and compared to a strong ML baseline classifier that is trained on a large amount of out-of-domain (OOD) data. The input to this classifier is embeddings from three networks trained on the VTD task, including embeddings from STRFNet [31], a WavLM-based network, and a pre-trained contrastive classifier that operates on x-vectors. These embeddings are then fused together using feed-forward layers, as shown in Fig. 3.3.

3.7 Preliminary Results

The infrastructure required to train a classifier with the IML paradigm has already been implemented, and preliminary experiments have been conducted on seven operating environments of the VTD task. This section compares training the contrastive classifier with the IML paradigm to training with a typical supervised paradigm on a large amount of out-of-domain (OOD) training data. The performance of the fusion model referenced in Section 3.6 is also included as a stronger baseline.

Experiments were run on the VTD task where AL samples were chosen randomly. Additional VTD

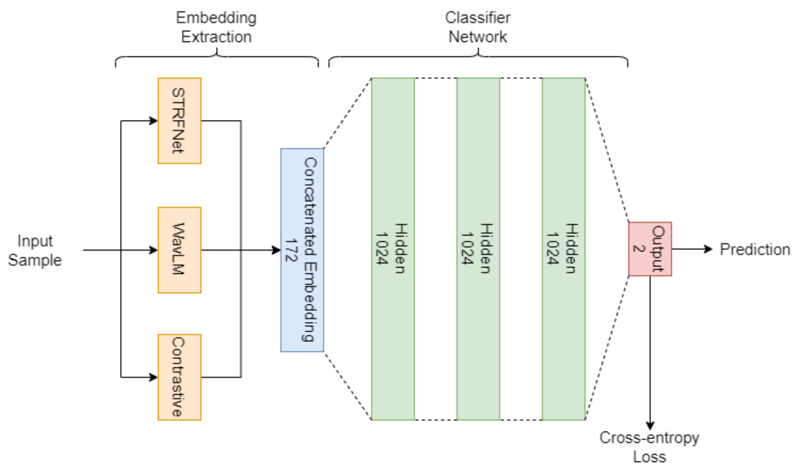

Figure 3.3: The architecture of the fusion classifier used as an ML baseline. Each of the embedding extractor blocks represents a neural network pre-trained on the VTD task.

experiments were also run using AL sampling methods based on Softmax score and negative energy certainty score. These methods are described in the following subsections. All numerical results are displayed in Table 3.1, and the trends across time are shown in Fig. 3.4. The vertical axis of Fig. 3.4 refers to the aggregated DCF accumulated across all sessions up to the corresponding session on the horizontal axis.

Architecture	Training paradigm	Mean training set size (hours)	DCF
Contrastive	ML	2292	0.5514
Fusion	ML	2292	0.0704
Contrastive	IML, Random AL	**1.2**	0.0707
Contrastive	IML, Softmax AL	**1.2**	**0.0521**
Contrastive	IML, Negative energy AL	**1.2**	0.0591

Table 3.1: Comparing typical supervised training to IML training with various AL sampling methods.

The IML feedback parameters for the results reported here are 100 bootstrap samples (50 from each class), and eight AL samples per session. CF was not used in these experiments for reasons discussed in Section 4.2.

The IML paradigm has yet to be evaluated on tasks other than VTD. Thus, I will apply the paradigm to both SED and SD according to the specifications above as part of the thesis work. The bulk of the proposed research follows in Chapter 4.

3.7.1 Softmax Score AL Sampling

The Softmax score is an uncertainty sampling method that uses Softmax output as a basic measure of a neural network classifier's confidence in its predictions. Since the output of the final Softmax activation

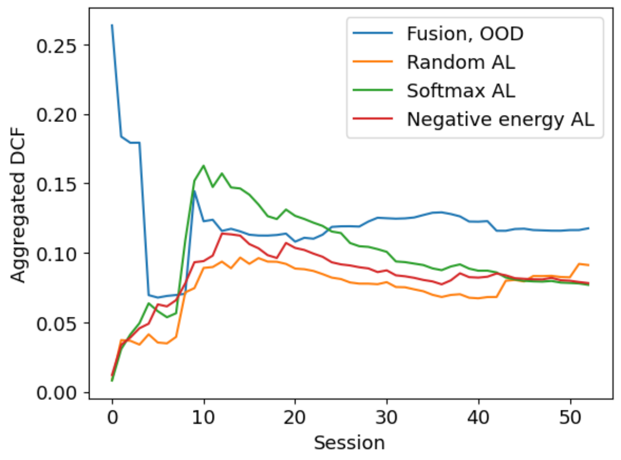

Figure 3.4: Aggregated DCF results using different AL sampling methods. Each line indicates the DCF accumulated over all sessions up to the session indicated on the horizontal axis, averaged over the first 53 sessions of seven environments.

layer of a classifier is a probability distribution across all possible classes, the distributions with a single clear maximum class probability are considered more confident, while distributions with maxima that are less clear are considered less confident. Hence, a confidence score based on Softmax output distributions can be defined for each data point as the maximum value of the distribution for that data point. Data points with low scores do not have clear maxima, indicating that the network was less confident in the decision [32].

3.7.2 Negative Energy Certainty Score AL Sampling

The negative energy certainty score is another uncertainty sampling method [33]. Here, rather than relying on the Softmax activation to normalize the network output into a probability distribution, raw network output from directly before the Softmax layer is used as a measure of energy produced by the network for a given input. During training, neural networks maximize the amount of energy produced by samples from the training distribution, but samples from outside that distribution are likely to yield less energy. As such, it is likely most useful to query the samples with the lowest energy, since those are the samples with which the network is least familiar.

Empirical studies show that negative energy certainty score outperforms Softmax score for OOD detection [33]. It also works quite well as an AL sampling method [34].

Chapter 4

Proposed Work

The IML paradigm acts as a framework under which many additional scientific contributions can be made. This chapter outlines the research topics related to IML that I intend to include in my thesis work.

4.1 Characterizing the ML Objective

As stated above, the objective of the IML paradigm is to jointly minimize the classification error and the amount of effort required of the domain expert. These two aspects are in opposition to one another, as more labeled data provided by the domain expert will inevitably lead to lower error, and less feedback will lead to higher error. Thus, optimizing IML training equates to finding the desired tradeoff between these two aspects.

A numerical measure of this objective has yet to be defined. An objective function must be defined in order to evaluate the IML training procedure. An objective function is also key to predicting model performance based on different parameter settings. This is especially important when deciding the size of the bootstrap corpus, and in determining when the classifier is accurate enough to be trusted by the domain expert.

This section proposes a simple objective function, as well as a mathematical model of the relationship between this objective, classification error rate, and the amount of labeled data required of the user.

4.1.1 Defining the IML Objective

The objective function for IML can be stated as in Eq. 4.1, where $\mathcal{L}$ is the overall objective to be minimized, $\mathcal{L}_E$ is the classification error, and $\mathcal{L}_F$ is the amount of required feedback. α is a weighting parameter that can take values between 0 and 1. This weighting is highly dependent on the sensitivity of the operation. As such, α should defined by the domain expert.

$$\mathcal{L} = \alpha\mathcal{L}_E + (1-\alpha)\mathcal{L}_F \tag{4.1}$$

4.1.2 Modeling Performance and Feedback

The objective function is useful as an evaluation metric. However, without considering the relationship between error and feedback, the objective could be arbitrarily minimized to zero by setting both $\mathcal{L}_E$ and $\mathcal{L}_F$ to zero. This, of course, is not possible because $\mathcal{L}_E$ and $\mathcal{L}_F$ are negatively correlated, which adds a constraint to the objective function.

The relationship between $\mathcal{L}_E$ and $\mathcal{L}_F$ can likely be approximated with a mathematical model. Conceptually, it must hold that when no feedback is provided, IML training will yield a model that achieves completely random performance since it has not been trained. It must also hold that when an infinite amount of feedback is provided, a sufficiently expressive classifier will achieve perfect performance. As such, the relationship between error and feedback might look something like Eq. 4.2. Here, ϕ denotes random performance (presumably 0.5 for binary classification tasks) and λ is the exponential rate at which error drops as feedback increases.

$$\mathcal{L}_E = \phi e^{-\lambda\mathcal{L}_F} \tag{4.2}$$

Weighted errors like DCF should also be considered. As such, the error rate can be rewritten in terms of a weighted combination of false negative errors (E_n) and false positive errors (E_p), like in Eq. 4.3. In the case of DCF, the weight parameter β would be set to 0.75.

$$\mathcal{L}_E = \beta E_n + (1-\beta)E_p \tag{4.3}$$

False negatives and false positives may be influenced differently by the task, parameter settings, class distribution, etc. Thus, it is advantageous to model the two error types separately, as below. Here, each error type is controlled by its own decay parameter, λ_n or λ_p.

$$E_n = \phi e^{-\lambda_n\mathcal{L}_F} \tag{4.4}$$

$$E_p = \phi e^{-\lambda_p\mathcal{L}_F} \tag{4.5}$$

Finally, putting everything together, the function relating error and feedback can be written as

$$\mathcal{L}_E = \beta\phi e^{-\lambda_n\mathcal{L}_F} + (1-\beta)\phi e^{-\lambda_p\mathcal{L}_F}. \tag{4.6}$$

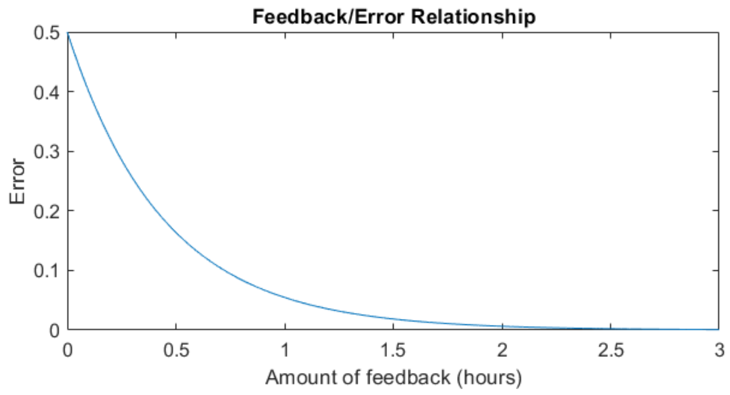

Figure 4.1: Example error vs. feedback curve modeled by Eq. 4.6.

An example curve with decay parameters calculated from experimental results is shown in Fig. 4.1. Here, β is assumed to be 0.75 since the error metric for VTD is DCF, and ϕ is set to 0.5. λ_n and λ_p are estimated based on the measured error rates and amount of feedback.

4.1.3 Optimizing the IML Objective

In IML, the primary tunable parameter is the amount of feedback to request. Since the classification error is now defined as a function of feedback, the objective can be rewritten by substituting Eq. 4.6 into Eq. 4.1, as follows:

$$\mathcal{L} = \alpha \left[\beta \phi e^{-\lambda_n \mathcal{L}_F} + (1-\beta) \phi e^{-\lambda_p \mathcal{L}_F} \right] + (1-\alpha)\mathcal{L}_F. \tag{4.7}$$

Combining the objective function with the performance-feedback model in this way is meaningful, since Eq. 4.7 can now be optimized by changing only the amount of feedback required. This is a differentiable function with a clear minimum, as shown in Fig. 4.2. This minimum, which equates to about five samples per hour, is much lower than the eight AL queries per hour that were requested in this example run.

4.1.4 Proposed Work

I propose the above objective function to evaluate the IML paradigm, as well as the underlying mathematical model which can be used to predict IML performance and estimate the optimal amount of feedback to request from the domain expert. As part of this proposed work, I intend to gather a wide range of performance results to verify this model and adjust it as needed. The accuracy of the mathematical model will be measured by goodness-of-fit for the collected data points.

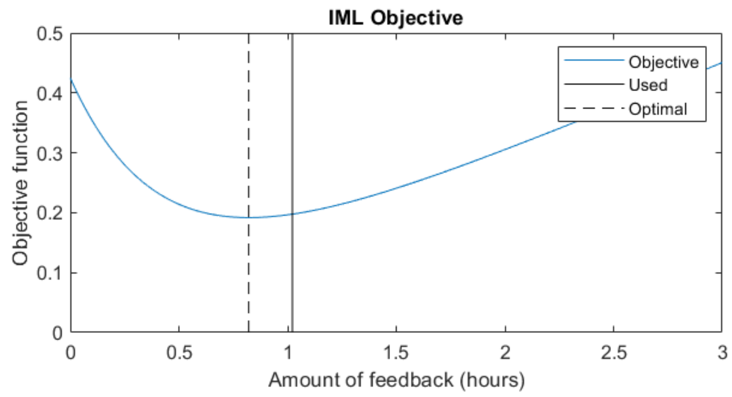

Figure 4.2: Objective function vs. feedback curve and suggested optimal feedback rate.

Once the mathematical model is verified, its practical applicability must be established. The model will be useful in a real-world setting if the λ parameters can be estimated without oracle information outside of what is provided via AL and CF. If the parameters can indeed be estimated based on the classification error on adaptation labels, it can be directly applied to estimate the ideal size of the bootstrap corpus and the ideal number of AL queries to pose per session during the adaptation cycle.

4.2 Controlling Errors with AL

One crucial missing piece of the IML puzzle is handling imbalance, both in the evaluation metric and in the data distribution. All audio detection tasks discussed in Section 3.4 use some form of weighted error function to punish either false negatives or false positives much more severely than the other. It is also often the case that the target class occurs relatively rarely in the data, which necessitates automatic detection methods to sort through the data in the first place.

The CF sampling piece of the IML pipeline contributes even more imbalance. While it does provide valuable annotations for adapting the classifier, the annotations are only for false positive (i.e., non-target) samples. This means that acquiring enough labeled target class samples to balance out the adaptation data fall squarely on AL sampling.

Most AL methods do not directly account for these imbalances, but there are a few rare exceptions. For example, [35] and [36] use AL to prune a labeled imbalanced dataset into an optimal balanced dataset. The weighted error problem is addressed in [37] by a special AL objective function referred to as Cost Overlapped Active Learning (COAL), which attempts to use regression to estimate the cost of each sample directly. Additional work even breaks down the assumption that the oracle is a perfect annotator and attempts to assign costs to certain samples accordingly [38].

4.2.1 Observations

AL Method	Metric	Overall Score
Random	DCF	0.0707
	FNR	0.0818
	FPR	0.0375
Softmax	DCF	**0.0521**
	FNR	**0.0601**
	FPR	0.0282
Negative Energy	DCF	0.0591
	FNR	0.0712
	FPR	**0.0227**

Table 4.1: FNR is higher than FPR in even the best cases.

Table 4.1 shows that, even for the AL sampling methods that achieve competitive scores for the VTD task, the FNR is always higher than the FPR, which is completely the opposite of the desired result. This is likely a side effect of the significant class imbalance in the VTD data. Nevertheless, it should be noted that not only is the FNR higher, the absolute number of false negative predictions is also consistently higher than the number of false positives. This indicates that there is something fundamental missing from the current AL formulation.

4.2.2 Preliminary Experiments

A massive class imbalance can be created by switching off the AL component of the pipeline, allowing labeled data to enter the adaptation pool only via bootstrap corpus creation and CF. As expected, the performance of this setup displayed in Table 4.2 is quite poor. While it does learn to reject non-target samples, it does not learn to accept target samples, resulting in a very high FNR and DCF.

Sampling method	DCF	FNR	FPR
AL Only	**0.0521**	**0.0601**	0.0282
CF Only	0.5446	0.7204	**0.0172**

Table 4.2: The effect of removing AL feedback.

4.2.3 Proposed Work

I propose an AL method that automatically adjusts for class imbalance in a way that optimizes classification error for the task and data at hand. This adjustment will be achieved by making the sample selection algorithm aware of the current class balance in the adaptation data pool.

This piece of proposed work will be accomplished in two parts. First, before the itself method can be developed, experiments with various distributions of target and non-target samples in the adaptation pool

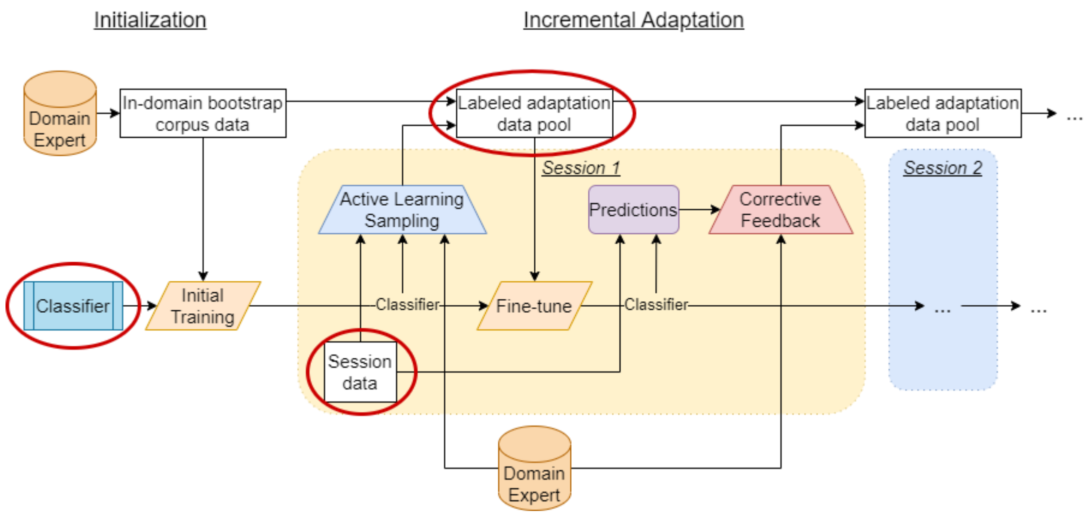

Figure 4.3: The IML pipeline with sources of change highlighted.

will need to be performed to show the effect of different balances on the two error types and overall error. Second, after the ideal balance has been illuminated by the experiments, the method will be developed with the ideal balance as a target. AL will be used to achieve this target by selecting more samples from the under-represented class based on the predictions of the model. For example, if more target class samples are needed, the majority of AL queries will be made about putative target-class samples, as predicted by the classifier.

The method will be evaluated by the IML objective function defined in Eq. 4.1 with a fixed α parameter. In other words, if the proposed AL method reduces classification error using the same amount or less feedback, the method will be considered a success.

4.3 Optimizing the Objective over Time

One of the unique aspects of IML is how it uses AL methods over an evolving set of data. Currently, this aspect of the training process has only been used as a method of gathering feedback. However, it has potential to become another significant advantage of IML if it is exploited properly.

This section describes the types of changes that occur during IML training, the tools available to address the changes, and proposed methods of measuring and addressing the changes.

4.3.1 Sources of Change in IML

There are three expected sources of change in the IML paradigm, each of which is highlighted in Fig. 4.3. Each type of change is described below.

Classifier Changes

The classifier begins as a neural network with randomly initialized weights. As it is introduced first to the bootstrap corpus and later to the adaptation data, the weights adjust to optimize the performance on the classifier.

Tracking when the classifier has reached a level of reliable performance is of practical importance for reasons discussed in Section 4.1. The performance of the model is also important to the AL sampling process, since the model generates the scores that are used to select which samples to query.

The capability of the classifier can be tracked as a function of the amount of adaptation data available. The average negative energy confidence score can also be used to estimate the general confidence of the classifier on the current session data or the available adaptation data.

Current Session Data Changes

The type of change that is the most detrimental to overall performance is the change in the distribution of data in the current session. Even within a single operating environment, the data distribution can change significantly enough to cause a large upswing in classification error.

Unsupervised methods for tracking data drift are common in CDA. These methods could be applied directly to the IML paradigm. However, methods for adapting the classifier when data drift is detected have not yet been developed for IML.

Adaptation Pool Changes

As the adaptation cycle progresses, more and more data are added to the adaptation data pool. More adaptation data often results in a more robust classifier. However, there may be cases where some sessions are similar to some previous sessions and less similar to others. In such cases, it may be beneficial to track the similarity between the data in the current session and the data in the adaptation pool. A simple metric for similarity would be the cosine similarity between feature vectors.

4.3.2 Points of Control in IML

There are two points of direct control in the IML pipeline, both of which are highlighted in Fig. 4.4 and further described below. All other pieces of the pipeline as it currently stands are affected either directly or indirectly by these two points of control. If the effects of these two points of direct control can be characterized, they can be automatically adjusted during training to optimize the process.

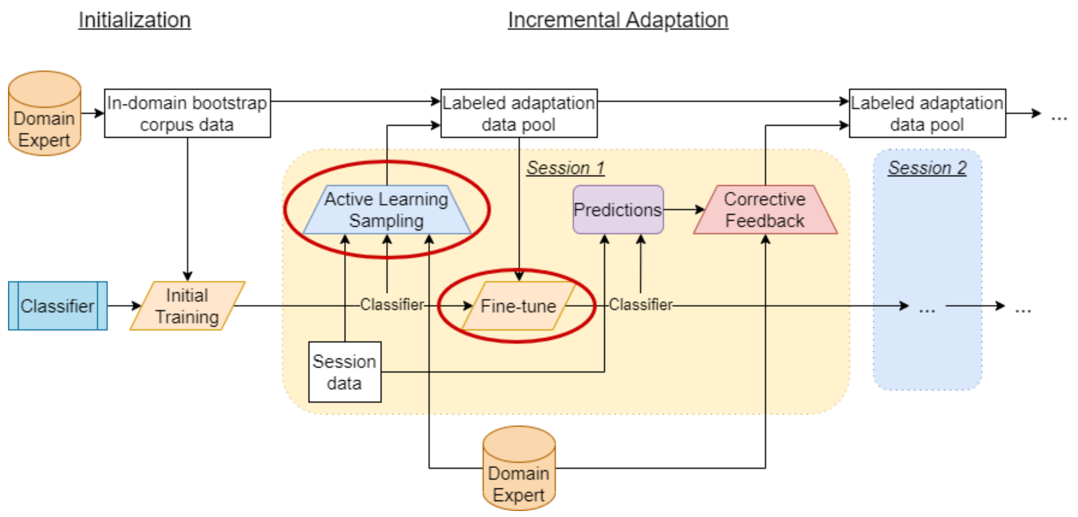

Figure 4.4: The IML pipeline with points of direct control highlighted.

AL Sampling

AL sampling can be controlled by the method used to determine which samples to query, as well as the number of samples queried. The observations that follow indicate ways in which the sampling method can be used to address changes in the classifier's performance, and how the sampling frequency can be adjusted to handle changes in the session data.

First, consider the inconsistency in Fig. 4.5, which compares the average DCF in seven environments over the first 53 sessions. While Softmax score-based sampling achieves the best performance overall, the negative energy score-based method shows superior performance for a number of sessions. The reasons for this discrepancy have yet to be explored thoroughly, but such a pattern could be exploited to optimize the IML objective more efficiently.

Second, consider Fig. 4.6, which shows the DCF of each session for a model only initialized with the bootstrap dataset (i.e., not adapted with AL or CF samples) in blue. Note that, while many sessions have very high DCF scores, there are also a substantial proportion of the sessions that perform well without any additional adaptation. As the classifier adapts to the operating environment, the number of future sessions that will not need to be adapted will continue to increase, as seen in the increased number of points below 0.05 DCF after just 20 sessions of adaptation. If the sessions that do not need adaptation can be identified reliably, AL queries for these sessions can be limited or skipped completely to reduce the amount of feedback required without sacrificing performance.

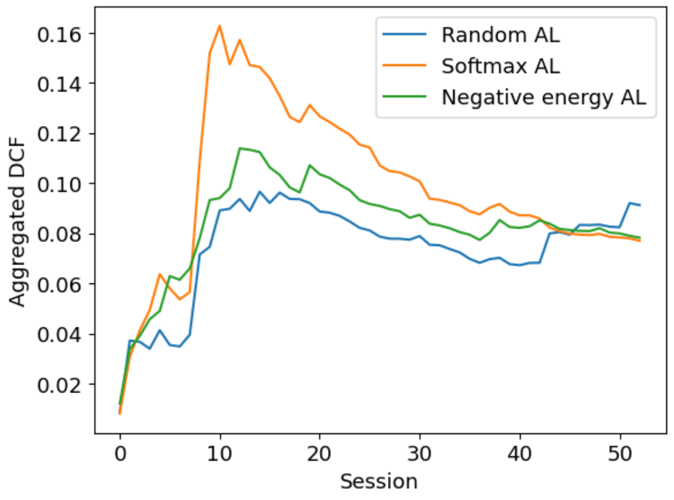

Figure 4.5: A comparison of different AL methods over time. No single method is always best.

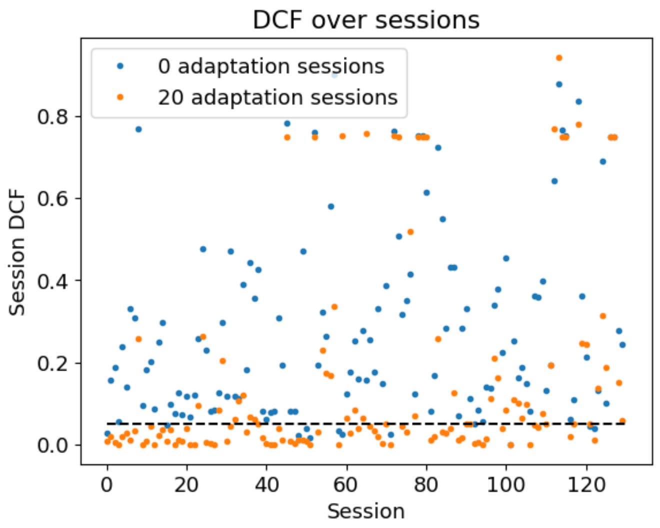

Figure 4.6: DCF over sessions from hotel mic 19 of the LB Corpus, with bootstrap only and with 20 sessions of adaptation.

Data Used for Fine-tuning

As the adaptation data pool grows, the options for what to do with that data increase. Assuming a highly expressive classifier, every one of the adaptation data points should be able to be classified correctly. However, depending on the distribution of the data in the current session, it may be beneficial to weight certain samples more heavily than others. For example, it would likely be advantageous to weight samples more heavily during fine-tuning if they are very similar to samples in the current session. It also may be useful to choose samples that will separate the classes more distinctly for the current session's data distribution.

4.3.3 Proposed Work

I propose the use of the identified points of control to address the potential changes during IML training. Specifically, the change in the classifier will be exploited by adjusting the AL sampling method over time, and the change in current session and adaptation data will be handled by adjusting the number of AL queries and the how the adaptation data is used during fine-tuning.

For the first step in understanding when to switch between AL sampling methods, an experiment will be performed in which all available AL sampling methods are used to choose adaptation samples at every AL step. Then, using oracle labels, the sampling method that achieved the best error at each step will be tracked. Based on trends found in the experimental results, a method based on classifier confidence will be developed to automatically estimate which sampling method to choose at each step. For example, experiments may indicate that the sum of negative energy scores across all samples in the session is a good indicator of model confidence for that session. In that case, the AL sampling method could be toggled based on the sum of the negative energy scores for data in the session at hand.

The number of AL queries needed for each session will be a function of how similar the session is to data logged in the adaptation data pool. This similarity can be measured using a clustering algorithm over both the embeddings of the labeled adaptation data and embeddings of the unlabeled session data. If the unlabeled embeddings mostly fall into the same clusters as the labeled embeddings, then it is likely that the current session data are similar to the adaptation data. In this case, fewer or no AL queries need to be made for this session; otherwise, the default maximum number of AL queries will be made. The correlation between data similarity and the number of queries required will be determined experimentally.

In the case that the temporal patterns revealed by the experiments proposed above are not obvious, a meta active learning algorithm may be advantageous to optimize the objective over time. Some previous work has been done on applying meta-learning methods to learn an optimal AL query selection method [39, 40, 41], while other work has applied reinforcement learning to a similar effect [42, 43]. Such approaches may be modified to fit the temporal changes that occur in IML.

Like the AL method proposed in Section 4.2.3, the proposed method will be evaluated by the IML objective function (Eq. 4.1) with a fixed α parameter.

4.4 Summary of Proposed Contributions

This chapter has outlined three proposed contributions as part of the IML paradigm that will comprise my thesis work. These contributions include:

1. A mathematical model that accurately characterizes the tradeoff between classifier performance and user workload in the IML paradigm. This includes running a series of IML experiments on a variety of tasks with a variety of different parameter settings to verify the model empirically. The model will be evaluated in terms of goodness-of-fit for the collected data points.

2. A method for optimizing weighted error metrics and handling significant class imbalance by controlling the false negative and false positive rates with AL sampling. The proposed method will be evaluated by the IML objective function with a pre-defined α parameter.

3. Methods for controlling aspects of AL sampling and adaptation to handle the identified changes that occur in the IML paradigm and further optimize the IML objective over time. These methods will also be evaluated using the IML objective function with a fixed weight.

Chapter 5

Timeline

Jun 2023	Thesis Proposal
Jun-Aug 2023	Characterizing the IML Objective
Sep-Nov 2023	Controlling Errors with AL
Dec 2023-Feb 2024	Optimizing the Objective over Time
Mar-Apr 2023	Thesis Writing
Apr 2024	Thesis Defense

References

[1] H. Lee, A. Saeed, and A. L. Bertozzi, "Active Learning of Non-Semantic Speech Tasks with Pretrained Models," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5. 2

[2] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual Lifelong Learning with Neural Networks: A Review," *Neural networks*, vol. 113, pp. 54–71, 2019. 4

[3] D. Kumaran and J. L. McClelland, "Generalization through the Recurrent Interaction of Episodic Memories: a Model of the Hippocampal System," *Psychological Review*, vol. 119, no. 3, p. 573, 2012. 5

[4] D. Lopez-Paz and M. Ranzato, "Gradient Episodic Memory for Continual Learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017. 5

[5] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming Catastrophic Forgetting in Neural Networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017. 5

[6] H. M. Gomes, A. Bifet, J. Read, J. P. Barddal, F. Enembreck, B. Pfharinger, G. Holmes, and T. Abdessalem, "Adaptive random forests for evolving data stream classification," *Machine Learning*, vol. 106, pp. 1469–1495, 2017. 6

[7] C. Zhang, Y. Cheng, P. Wei, H. He, and J. Chen, "CENet: Consolidation-and-Exploration Network for Continuous Domain Adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3426–3432. 6

[8] J. Hoffman, T. Darrell, and K. Saenko, "Continuous Manifold Based Adaptation for Evolving Visual Domains," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 867–874. 6

[9] B. Settles, "Active Learning Literature Survey," 2009. 6

[10] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A Survey of Deep Active Learning," *ACM computing surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021. 6

[11] D. Lewis and W. Gale, "A sequential algorithm for training text classifiers," in *SIGIR'94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, 1994, pp. 3–12. 7

[12] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, vol. 2, no. Nov, pp. 45–66, 2001. 7

[13] T. He, X. Jin, G. Ding, L. Yi, and C. Yan, "Towards better uncertainty sampling: Active learning with multiple views for deep convolutional neural network," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2019, pp. 1360–1365. 7

[14] J. T. Ash, C. Zhang, A. Krishnamurthy, J. Langford, and A. Agarwal, "Deep batch active learning by diverse, uncertain gradient lower bounds," *arXiv preprint arXiv:1906.03671*, 2019. 7

[15] B. Settles, M. Craven, and S. Ray, "Multiple-instance Active Learning," *Advances in Neural Information Processing Systems*, vol. 20, 2007. 7

[16] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active Learning with Statistical Models," *Journal of artificial intelligence research*, vol. 4, pp. 129–145, 1996. 7

[17] T. Zhang and F. Oles, "A Probability Analysis on the Value of Unlabeled Data for Classification Problems," in *ICML*. Stanford, US, 2000, pp. 1191–1198. 7

[18] B. Settles and M. Craven, "An Analysis of Active Learning Strategies for Sequence Labeling Tasks," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2008, pp. 1070–1079. 7

[19] N. Roy and A. McCallum, "Toward Optimal Active Learning through Sampling Estimation of Error Reduction," in *International Conference on Machine Learning (ICML)*. Morgan Kaufmann, 2001, pp. 441–448. 7

[20] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining Active Learning and Semi-supervised Learning Using Gaussian Fields and Harmonic Functions," in *ICML 2003 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, vol. 3, 2003. 7

[21] Y. Guo and R. Greiner, "Optimistic Active-Learning Using Mutual Information," in *IJCAI*, vol. 7, 2007, pp. 823–829. 7

[22] M. Lindsey, T. Vuong, and R. M. Stern, "Unsupervised Voice Type Discrimination Score Adaptation Using X-Vector Clusters," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5. 11

[23] A. Politis, A. Mesaros, S. Adavanne, T. Heittola, and T. Virtanen, "Overview and Evaluation of Sound Event Localization and Detection in DCASE 2019," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 684–698, 2021. 12

[24] M. Cartwright, J. Cramer, A. E. M. Mendez, Y. Wang, H.-H. Wu, V. Lostanlen, M. Fuentes, G. Dove, C. Mydlarz, J. Salamon, O. Nov, and J. P. Bello, "SONYC Urban Sound Tagging (SONYC-UST): a multilabel dataset from an urban acoustic sensor network," Sep. 2020, This work is supported by National Science Foundation award 1544753. [Online]. Available: https://doi.org/10.5281/zenodo.3966543 12

[25] J. Yamagishi, M. Todisco, M. Sahidullah, H. Delgado, X. Wang, N. Evans, T. Kinnunen, K. A. Lee, V. Vestman, and A. Nautsch, "ASVspoof 2019: The 3rd Automatic Speaker Verification Spoofing and Countermeasures Challenge Database," 2019. 13

[26] ——, "ASVspoof 2019: Automatic Speaker Verification Spoofing and Countermeasures Challenge Evaluation Plan," *ASV Spoof*, 2019. 13

[27] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, pp. 539–546 vol. 1. 13

[28] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN Embeddings for Speaker Recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333. 13

[29] Y. Wu, K. Chen, T. Zhang, Y. Hui, T. Berg-Kirkpatrick, and S. Dubnov, "Large-scale Contrastive Language-Audio Pretraining with Feature Fusion and Keyword-to-caption Augmentation," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5. 13

[30] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "WavLM: Large-scale Self-supervised Pre-training for Full Stack Speech Processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022. 13

[31] T. Vuong, Y. Xia, and R. M. Stern, "Learnable Spectro-temporal Receptive Fields for Robust Voice Type Discrimination," in *Interspeech 2020*. ISCA, Oct. 2020, pp. 1957–1961. 14

[32] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential Deep Learning to Quantify Classification Uncertainty," *Advances in neural information processing systems*, vol. 31, 2018. 16

[33] W. Liu, X. Wang, J. Owens, and Y. Li, "Energy-based Out-of-distribution Detection," *Advances in neural information processing systems*, vol. 33, pp. 21 464–21 475, 2020. 16

[34] X. Wang and J. Yamagishi, "Investigating Active-Learning-Based Training Data Selection for Speech Spoofing Countermeasure," in *2022 IEEE Spoken Language Technology Workshop (SLT)*, 2023, pp. 585–592. 16

[35] S. Ertekin, J. Huang, and C. L. Giles, "Active Learning for Class Imbalance Problem," in *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007, pp. 823–824. 20

[36] S. Ertekin, J. Huang, L. Bottou, and L. Giles, "Learning on the Border: Active Learning in Imbalanced Data Classification," in *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, 2007, pp. 127–136. 20

[37] A. Krishnamurthy, A. Agarwal, T.-K. Huang, H. Daumé III, and J. Langford, "Active Learning for Cost-sensitive Classification," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1915–1924. 20

[38] P. Donmez and J. G. Carbonell, "Proactive learning: Cost-sensitive active learning with multiple imperfect oracles," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, 2008, pp. 619–628. 20

[39] S. Ravi and H. Larochelle, "Meta-Learning for Batch Mode Active Learning," 2018. [Online]. Available: https://openreview.net/forum?id=r1PsGFJPz 27

[40] V. E. Martins, A. Cano, and S. B. Junior, "Meta-learning for dynamic tuning of active learning on stream classification," *Pattern Recognition*, vol. 138, p. 109359, 2023. 27

[41] K. Konyushkova, R. Sznitman, and P. Fua, "Learning Active Learning from Data," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates,

Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/8ca8da41fe1ebc8d3ca31dc14f5fc56c-Paper.pdf 27

[42] K. Pang, M. Dong, and T. Hospedales, “Meta-Learning Transferable Active Learning Policies by Deep Reinforcement Learning,” 2018. [Online]. Available: https://openreview.net/forum?id=HJ4IhxZAb 27

[43] M. Fang, Y. Li, and T. Cohn, “Learning How to Active Learn: A Deep Reinforcement Learning Approach,” *arXiv preprint arXiv:1708.02383*, 2017. 27