Journal of Machine Learning Research 23 (2022) 1-29 Submitted 4/21; Revised 12/21; Published 2/22

IALE: Imitating Active Learner Ensembles

Christoffer Löffler CHRISTOFFER.LOEFFLER@FAU.DE
Machine Learning and Data Analytics Lab
Friedrich-Alexander University Erlangen-Nürnberg (FAU)
Carl-Thiersch-Straße 2b, 91052, Erlangen, Germany

Christopher Mutschler CHRISTOPHER.MUTSCHLER@IIS.FRAUNHOFER.DE
Fraunhofer IIS
Fraunhofer Institute for Integrated Circuits IIS
Nuremberg, Germany

Editor: Andreas Krause

Abstract

Active learning prioritizes the labeling of the most informative data samples. However, the performance of active learning heuristics depends on both the structure of the underlying model architecture and the data. We propose IALE[1], an imitation learning scheme that imitates the selection of the best-performing expert heuristic at each stage of the learning cycle in a batch-mode pool-based setting. We use DAgger to train a transferable policy on a dataset and later apply it to different datasets and deep classifier architectures. The policy reflects on the best choices from multiple expert heuristics given the current state of the active learning process, and learns to select samples in a complementary way that unifies the expert strategies. Our experiments on well-known image datasets show that we outperform state of the art imitation learners and heuristics.

Keywords: active learning, deep neural networks, imitation learning, dataset aggregation, transferable policy

1. Introduction

The high performance of deep learning on various tasks from computer vision (Voulodimos et al., 2018) to natural language processing (NLP) (Barrault et al., 2019) also comes with a few disadvantages. One of the major drawbacks is the large amount of labeled training data they require. Obtaining such data is expensive and time-consuming and often requires domain expertise (Löffler et al., 2020).

Active Learning (AL) is an iterative process where during every iteration an oracle (e.g., a human) is asked to label the most informative unlabeled data sample(s). In *pool-based* AL all data samples are available (while most of them are unlabeled). In *batch-mode* pool-based AL, we select unlabeled data samples from the pool in acquisition batches greater than 1. Batch-mode AL decreases the number of AL iterations required and makes it easier for an oracle to label the data samples (Settles, 2009). As a selection criteria we usually need to quantify how informative a label for a particular sample is. Well-known criteria include heuristics such as model uncertainty (Gal et al., 2017; Roth and Small, 2006; Wang

1. `IALE` is pronounced /eɪl/.

and Shang, 2014; Ash et al., 2020), data diversity (Sener and Savarese, 2018), query-by-committee (Beluch et al., 2018), and expected model change (Settles et al., 2008). As ideally we label the most informative data samples at each iteration, the performance of a machine learning model trained on a labeled subset of the available data selected by an AL strategy is better than that of a model that is trained on a randomly sampled subset of the data.

Besides the above mentioned, in the recent past several other data-driven AL approaches emerged. Some are modelling the data distributions (Mahapatra et al., 2018; Sinha et al., 2019; Tonnaer, 2017; Hossain et al., 2018) as a pre-processing step, or similarly use metric-based meta-learning (Ravi and Larochelle, 2018; Contardo et al., 2017) as a clustering algorithm. Others focus on the heuristics and predict the best suitable one using a multi-armed bandits approach (Hsu and Lin, 2015). Recent approaches that use reinforcement learning (RL) directly learn strategies from data (Woodward and Finn, 2016; Bachman et al., 2017; Fang et al., 2017). Instead of pre-processing data or dealing with the selection of a suitable heuristic they aim to learn an optimal selection sequence on a given task.

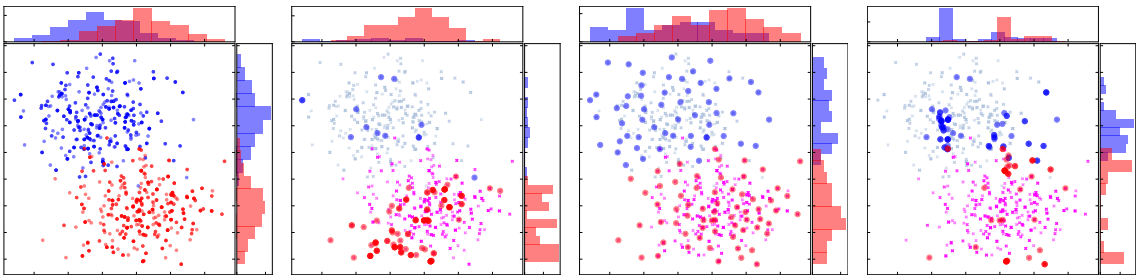
However, the RL approaches not only require a huge amount of samples they also do not resort to existing knowledge, such as potentially available AL heuristics. Moreover, training the RL agents is usually time-consuming as they are trained from scratch. Hence, when only few labeled training data and a potent algorithmic expert are available imitation learning (IL) helps. IL trains, i.e., *clones*, a policy to transfer the expert to the related few data problem. While IL mitigates some of the aforementioned issues, previous approaches are still limited (including that of Liu et al. (2018)), e.g., by their limited expressiveness of the *state* representations, their computational efficiencies, their non-arbitrary acquisition sizes, and their lack of complementary experts. They were also so far only evaluated on NLP tasks.

We propose `IALE`, that is based on imitation learning and that makes use of a diverse set of experts from different heuristic families, i.e., uncertainty, diversity, expected model-change, and query-by-committee, in a batch-mode AL setting with *arbitrary* acquisition sizes. Our policy extends previous work (see Section 2) by learning at which stage of the AL cycle which of the available strategies performs best, based on a more expressive state, that allows a powerful introspective view into the classifier model to better assess its confidence. We use Dataset Aggregation (DAgger) to train a robust and transferable policy and apply it to other problems from similar domains (see Section 3). We show that we can (1) train a policy on image datasets such as MNIST, Fashion-MNIST, Kuzushiji-MNIST, Extended MNIST, CIFAR and SVHN, (2) transfer the policy between them, and (3) even transfer the policy between different classifier architectures (see Section 4).

2. Related Work

Next to the AL approaches for traditional ML models (Settles, 2009) also ones applicable to deep learning have been proposed (Gal et al., 2017; Sener and Savarese, 2018; Beluch et al., 2018; Settles et al., 2008; Ash et al., 2020). Below we discuss AL strategies that are trained on data.

Generative Models. Explicitly modeled data distributions capture the *informativeness* that can be used to select samples based on diversity. VAAL (Sinha et al., 2019) is a pool-based semi-supervised AL method, where a discriminator discriminates between labeled and unlabeled samples using the latent representations of a variational autoencoder. The



(a) Data distribution (b) *Entropy* (c) *CoreSet* (d) *Our method*

Figure 1: Active learning on two normal distributions in a 2D feature space. Fig. (1a) shows data and histograms. The remaining plots show labelled and unlabelled data, and histograms of labelled samples for different acquisition strategies. We show the state after training an MLP (two layers with 16 units and ReLU activation) via AL (20 initially labelled samples, acquiring 10 labels per iteration, 10 times). We present (b) *Entropy* sampling, (c) *CoreSet* sampling, and (d) `IALE`, that imitates the other two, see Section. 3 for details.

representations are used to pick the most diverse and representative data points (Tonnaer, 2017). Mirza and Osindero (2014) use a conditional generative adversarial network to generate samples with different characteristics from which the most informative are selected using the uncertainty measured by a Bayesian neural network (Kendall and Gal, 2017; Mahapatra et al., 2018). Such approaches are similar to ours (as they capture dataset properties) but instead we model the dataset implicitly and infer a selection heuristic via imitation.

Metric Learning. Metric learners such as the one proposed by Ravi and Larochelle (2018) use a set of statistics calculated from the clusters of un-/labeled samples in a Prototypical Network's (Snell et al., 2017) embedding space, or learn to rank (Li et al., 2020) large batches. Such statistics use distances (e.g., Euclidean distance) or are otherwise converted into class probabilities. Two MLPs predict either a quality or diversity query selection using backpropagation and the REINFORCE gradient (Mnih and Rezende, 2016). While they rely on statistics over the classifier's embedding and explicitly learn two strategies (quality and diversity) we use a richer state and are not constrained to specific strategies.

Meta learning. A similar field is *learning-to-learn*. Especially optimization-based meta learning, that aims to improve or discover learning algorithms (Hochreiter et al., 2001), potentially leads to alternative and fast converging learning algorithms for deep learning in few data (or few-shot) settings. Methods such as the LSTM Meta Learner (Ravi and Larochelle, 2017), which uses an LSTM to predict a network's parameter updates, the recurrent neural network-based approach by Chen et al. (2017), which learns to optimize black-box functions within a fixed horizon, and model-agnostic meta-learning (MAML) (Finn et al., 2017), which produces a well performing parameter initialization for the (task-specific) differentiating fine-tuning stage, are exemplary approaches. However, not only are these methods often computationally expensive to train, e.g., due to MAML's meta-gradient updates (Nichol et al., 2018). They also to tend to overfit on hard tasks (Jamal and Qi, 2019) or even fail to converge for ambiguous small tasks (Finn et al., 2018). Moreover, we do not require back-propagation through time (as Ravi and Larochelle (2017) do) with its

limiting time horizon (Mishra et al., 2018; Chen et al., 2017), but instead rely on gradient descent in combination with a high capacity state for our learned policy. This allows us to generalize to nearly arbitrary horizons including larger ones than seen during policy training.

Reinforcement Learning (RL). The AL cycle can be modeled as a sequential decision making problem. Woodward and Finn (2016) propose a stream-based AL agent based on memory-augmented neural networks where an LSTM-based agent learns to decide whether to predict a class label or to query the oracle. Matching Networks (Bachman et al., 2017) extensions allow for pool-based AL. Fang et al. (2017) use Deep Q-Learning in a stream-based AL scenario for sentence segmentation. In contrast to them we consider batch-mode AL with acquisition sizes ≥ 1 and work on a pool-setting instead of a stream-setting. While Bachman et al. (2017) propose a strategy to extend the RL-based approaches to a pool setting, they still do not work on batches. Instead, we allow batches of arbitrary acquisition sizes. Konyushkova et al. (2017) formulate AL as a regression task for a greedy label acquisition, that predicts the expected reduction of the classification error for each sample, and that is trained on either synthetic or real data. They use a Random Forest classifier, with features like predicted probability and forest variance, for binary classification and batch-sizes of one. Follow-up work (Konyushkova et al., 2018) replaces the greedy approach with a Q-Learning-based RL agent. Casanova et al. (2020) propose a DQN-based extension of Konyushkova et al. (2018)'s method, that learns to sample image regions for a semantic image segmentation task, focusing on classes that are underrepresented in the training dataset. Their work is specifically aimed at selecting relevant regions in images to optimize the classes' mean intersection of union. Our work focuses on different problems for learning AL, as we investigate useful sample relevance features for AL especially from deep neural networks, learn a unified AL heuristic from existing experts, and investigate the transfer of the policy between real datasets of multi-class problems with larger batch-sizes, on more complex datasets and transfers between classifier architectures. Fan et al. (2018) propose a meta-learning approach that trains a student-teacher pair via RL. The teacher optimizes *data teaching* by selecting labeled samples that let the student learn faster. In contrast, our method learns to selects samples from an unlabeled pool, i.e., in a missing target scenario. The teacher-student analogy is similar to our approach, however, the objective, method and available (meta-)data to learn a good teacher (policy) are considerably different.

Multi-armed Bandit (MAB). Baram et al. (2004) treat the online selection of AL heuristics from an ensemble as the choice in a multi-armed bandit problem. COMB uses the known EXP4 algorithm to solve it, and ranks AL heuristics according to a semi-supervised maximum entropy criterion (Classification Entropy Maximization) over the samples in the pool. Building on this, Hsu and Lin (2015) learn to select an AL strategy for an SVM-classifier and use importance-weighted accuracy extension to EXP4 that better estimates the performance of each AL heuristic improvement as an unbiased estimator for the test accuracy. Furthermore, they reformulate the MAB setting so that the heuristics are treated as the bandits where the algorithm selects the one with the largest performance improvement (in contrast to COMB's formulation where the unlabeled samples are treated as bandits). Chu and Lin (2016) extend Hsu and Lin (2015) to a setting where the selection of AL heuristics is based on a linear weighting, aggregating *experience* over multiple datasets. They adapt the semi-supervised reward scheme from Hsu and Lin (2015) to work with their deterministic queries. Instead of selecting from a set of available heuristics, we propose the learning of a

unified AL policy. This allows our policy model to learn an *interpolation* between batches of samples proposed by single heuristics also exploiting the deep network classifier's internal state.

Imitation Learning (IL). Imitation learning methods such as DAgger (Ross et al., 2011) can also be used to train an AL policy. For instance, Liu et al. (2018) propose a follow-the-leader approach that selects samples that improve classifier accuracy. During policy training they *roll out* a few possible acquisitions (using a small random pool subset) and retrain the classifier on each sample independently to infer *preference scores*. However, this not only leads to sub-optimal selections, it also requires an expensive re-training per sample in the roll-out. In contrast to them, we explore an alternative way for using IL, as `IALE` imitates an ensemble of a wide variety of AL heuristics to learn a unified AL strategy. Our *state* consists of novel features for *introspection*, such as gradients inferred from proxy-labels, as similarily proposed by Ash et al. (2020). Hence, `IALE` is well suited for deep learning (efficient inference, fast convergence), even compared to classical AL baselines.

3. IALE: Imitating Active Learner Ensembles

`IALE` learns an AL sampling strategy from *multiple experts* in a *pool-based* setting by imitating their behavior. We train a policy with data consisting of states (that encode, e.g., labeled and unlabeled sample distributions, uncertainty, and gradient signals) and best expert actions (i.e., samples selected for labeling) collected over the AL cycles. Hence, our policy learns with options, where each expert's (potentially sub-optimal) selection is an option it may choose to learn, according to their rank. Analogously, our approach is similar to a distillation of the experts. The policy is then applied on a different task. To discover states that are unlikely to be produced by the experts, DAgger (Ross et al., 2011) balances exploration (via the current policy) and exploitation (via the AL experts) to collect a large set of states and actions. We train the policy network on all the previous states and actions after each AL iteration.

3.1 Background

In pool-based AL we train a model M on a dataset $\mathcal{D}$ by iteratively labeling data samples. Initially, M is trained on a small amount of labeled data $\mathcal{D}_{\text{lab}}$ randomly sampled from the dataset. The rest of the data is considered as the unlabeled data pool $\mathcal{D}_{\text{pool}}$, i.e., $\mathcal{D} = \mathcal{D}_{\text{lab}} \cup \mathcal{D}_{\text{pool}}$. From that point onwards during the AL iterations a subset of $\mathcal{D}_{\text{sel}}$ is selected from $\mathcal{D}_{\text{pool}}$ by using an acquisition function $a(M, \mathcal{D}_{\text{pool}})$. The data is labeled and then removed from $\mathcal{D}_{\text{pool}}$ and added to $\mathcal{D}_{\text{lab}}$. The size of $\mathcal{D}_{\text{sel}}$ is based on the acquisition size acq (>1 for batch-mode AL). The AL cycle continues until a labeling budget of $\mathcal{B}$ is reached. M is retrained after each acquisition to evaluate the performance boost with respect to the increased labeled dataset only (and not the additional training time).

The acquisition function a uses heuristics on the trained model M to select the most informative data samples from $\mathcal{D}_{\text{pool}}$. For deep AL those include uncertainty-based *MC-Dropout* (Gal et al., 2017), query-by-committee-based *Ensembles* (Beluch et al., 2018), data diversity-based *CoreSet* (Sener and Savarese, 2018), gradient-based *BADGE* (Ash et al., 2020), and soft-max-based *Confidence-* or *Entropy*-sampling (Wang and Shang, 2014).

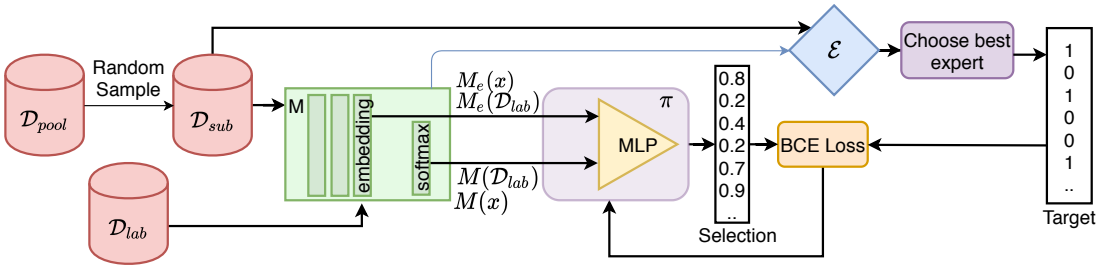


Figure 2: Training π to imitate experts $\mathcal{E}$: (1) we pass samples from $\mathcal{D}_{\text{sub}}$ and $\mathcal{D}_{\text{lab}}$ through the current classifier M; (2) the embeddings and the predictions are input to π, whose output vector is compared with the target vector predicted by the best expert; (3) we calculate a loss and back-propagate the error through π; (4) we extend the labeled pool data $\mathcal{D}_{\text{lab}}$ by $\mathcal{D}_{\text{sel}}$ and retrain M.

MC-Dropout uses a Monte-Carlo inference scheme based on a *dropout* layer to approximate the model's predictive uncertainty (Gal and Ghahramani, 2016) and then uses these values to select the most uncertain samples (Gal et al., 2017). *Ensembles* (Beluch et al., 2018) model predictive uncertainty using a committee of N classifiers, initialized with different random seeds. However, while at inference time we need to run only N forward-passes per sample (compared to *MC-Dropout* performing two dozens or more Monte-Carlo passes), the training of N-1 additional deep models can become prohibitively expensive in many use-cases. *CoreSet* (Sener and Savarese, 2018) aims to select diverse samples by solving the k-centers problem on the classifier's embeddings. This involves minimizing the distance between each of the unlabeled data samples to its nearest labeled samples. *BADGE* determines the magnitudes of the gradients in a batch using proxy-labels and selects samples by uncertainty and diversity. Soft-max-based heuristics (*Confidence*- and *Entropy*-sampling) use predictive uncertainty and are computationally lightweight at lower AL performance (Gal and Ghahramani, 2016; Ash et al., 2020). *Confidence* selects the samples with the lowest class probability and *Entropy* the ones with largest entropy of their probability distribution.

3.2 Learning Multiple Experts

Instead of using specific heuristics we propose to learn the acquisition function using a policy network. Once the policy is trained on a source dataset, it has learned a unified active learning heuristic, and can be applied to different target datasets.

Figure 1 illustrates the approach with two-dimensional Gaussian distributions. After ten acquisitions of ten new samples each strategy has sampled a distinct set of labeled data. While uncertainty-based *Entropy* mostly samples from only one class, the diversity-based *CoreSet* covers a more representative set over the data and selects data from the whole distribution. Our approach was trained to imitate both methods. Figure 1d shows that it acquired diverse samples (in this example at a ratio of 60 to 40), and samples especially along the decision boundary, reaching higher accuracy than any of the single baselines alone. The learned strategy hence combines the advantages of both imitated methods.

Figure 2 sketches the imitation learning framework. The policy network π is a Multi-Layer Perceptron (MLP) trained to predict the *usefulness* of labeling samples from the

unlabeled data pool $\mathcal{D}_{\text{pool}}$ for training the model M, similar to an AL acquisition function. As input the policy network takes the current *state*, that encodes, e.g., M's information on learned representations $M_e(x)$ and $M_e(\mathcal{D}_{\text{lab}})$, as well as its predictive uncertainty derived from $M(x)$, $M(\mathcal{D}_{\text{lab}})$, and label information from $\mathcal{D}_{\text{lab}}$. We use the predictions $M(x)$ to enrich the state with *pseudo-label* gradient signals to guide the policy's decisions later on[2]. Our state representation significantly extends previous work (Contardo et al., 2017; Konyushkova et al., 2017; Liu et al., 2018; Casanova et al., 2020) and adds novel features that allow for model introspection. The policy π then outputs the *action* to be taken at each step. *Action* here refers to an AL acquisition, i.e., which of the unlabeled data samples should be labeled and added to the training data. π learns the best *actions* from a set of experts $\mathcal{E}$ which predict the best actions for a given AL state, and thus learns its own *complementary* strategy. A subset of the pool dataset $\mathcal{D}_{\text{sub}}$ with size n is used instead of the whole pool dataset at each active learning iteration for training the policy.

States. As π uses the state information to make decisions, a state s should be maximally compact but still unique, i.e., different *situations* should have a different state encoding, and they have to allow to predict M's expected improvement. Our state encoding uses three types of information: (1) M's learned representations, (2) M's predictive uncertainty, and (3) M's gradient signals for unlabeled samples. We distinguish the state's parameters as either derived from the labeled or from the unlabeled pool. Together, these parameters form a minimal but comprehensive description of a model's state at each step of the AL-cycle.

For **labeled samples** the following parameters encode M's *learned representations* and *predictive uncertainty*:

- The embedded samples' mean $\mu\big(M_e(\mathcal{D}_{\text{lab}})\big)$: the embedding M_e of a sample by M is the output of the final layer (i.e., the layer before the soft-max layer in case of classification), see Figure 2. The size of this representation is independent of the (growing) size of $\mathcal{D}_{\text{lab}}$ and thus will not become a computational bottle-neck.

- The ground-truth empirical distribution of class labels

$$\vec{e}_{\mathcal{D}_{\text{lab}}} = \left(\frac{\sum_{y\in\mathcal{D}_{\text{lab}}} 1[y==0]}{|\mathcal{D}_{\text{lab}}|}, .., \frac{\sum_{y\in\mathcal{D}_{\text{lab}}} 1[y==i]}{|\mathcal{D}_{\text{lab}}|}\right),$$

  which is a normalized vector of length i, i.e., the number of classes, with percentage of occurrence per class using the labels of the already acquired data samples.

- M's predicted distribution of class labels for the labeled data $\vec{e}_{M(\mathcal{D}_{\text{lab}})}$, i.e., a normalized vector as before but with predicted class labels instead of ground-truth.

We encode the predictive uncertainty by including both the ground truth and the predicted empirical distribution. The policy can base its decisions on the model M's prediction errors, e.g., by detecting wrong predictions of already labeled samples and decide to acquire more similar samples.

For **unlabeled samples** (n data samples in $\mathcal{D}_{\text{sub}}$), we encode the information that is necessary to help π acquire more relevant samples. First, we calculate M's representation for each data sample $x_i \in \mathcal{D}_{\text{sub}}$, i.e., its embedding $M_e(x_i)$ in the same embedding space as

2. see Eq. 1 for an exact definition of the state.

the already labeled samples. Second, we also predict each sample's label $M(x_i)$. Finally, we encode gradient signals as the gradients of unlabeled data provide a powerful view due to its effect on the classifier model (as they encode M's expected change directed towards the steepest learning steps). Although their calculation usually requires labeled samples we can still approximate them using proxy-labels (Ash et al., 2020). We define a proxy-label $\hat{y}$ (i.e., the one that has the highest class probability for $M(x_i)$). Then, the gradient's magnitude and direction at the embedding layer describes the model's uncertainty and its expected change. We thus capture gradient information at the embedding layer as $g(M_e(x_i))$ and encode it as part of the state.

In summary, the state enables the policy to learn to select samples (1) where the model is uncertain (i.e., where it predicts the wrong labels), (2) where the model might gain most information (i.e., the gradient's magnitude is large), and (3) to learn to select samples that increase the labeled pool's diversity (i.e., to acquire less well-represented samples, using the label statistics and the learned representations). Hence, we describe a state s as follows:

$$s := \Big[\mu(M_e(\mathcal{D}_{\text{lab}})), \vec{e}_{\mathcal{D}_{\text{lab}}}, \vec{e}_{M(\mathcal{D}_{\text{lab}})}, \begin{bmatrix} M_e(x_0) \\ \vdots \\ M_e(x_n) \end{bmatrix}, \begin{bmatrix} M(x_0) \\ \vdots \\ M(x_n) \end{bmatrix}, \begin{bmatrix} g(M_e(x_0)) \\ \vdots \\ g(M_e(x_n)) \end{bmatrix}\Big] \tag{1}$$

Actions. In our approach, actions are essentially the resulting selections from acquisition functions, that π learns to imitate. The ground truth actions (selections) provided by the experts are binary vectors of length k, where a 1 at index i means that x_i should be selected for labeling. We may think of the experts are policies themselves that only have a fixed acquisition function. `IALE`'s acquisition function, on the other hand, uses its neural network's prediction to select samples. The output of the MLP is analogously a vector with a *desirability score* ρ_i for each unlabeled sample from $\mathcal{D}_{\text{sub}}$, i.e., $\rho_i := \pi(s_i)$, from which we choose the highest ranked samples. This results in a binary selection vector $\vec{v} = (\rho_0, \cdots, \rho_n)$ with $\sum_{i=0}^{n} \vec{v}_i = \text{acq}$. We use a *binary cross entropy loss* to update π's weights:

$$\mathcal{L}(\rho, \vec{t}) = -\sum_{i=0}^{n} \vec{t}_i \log(\rho_i) - (1 - \vec{t}_i) \log(1 - \rho_i), \tag{2}$$

where $\vec{t}$ is the target vector provided by the *best* expert (similar to a greedy multi-armed bandit approach (Hsu and Lin, 2015)), guiding π towards the best expert's suggestion.

Our IL-based approach uses the experts to turn AL into a supervised learning problem, i.e., the action of the best expert becomes the label for the current state s. From all the experts $\mathcal{E}$ we determine the best one by letting all of them select samples for labeling, and then rank their performance using temporary models trained for each expert. Our choice of AL heuristics for the set of experts $\mathcal{E}$ includes particular types but is arbitrarily extendable. Using *MC-Dropout*, *Ensemble*, *CoreSet*, *BADGE*, *Confidence* or *Entropy* allows us to only minimally modify the classifier model M, e.g., we add dropout at inference to use *MC-Dropout*. π aims to learn certain derived properties from s, such as model uncertainty.

Our hypothesis is that π imitates the best suitable heuristic for each phase of the AL cycle, i.e., starting with relying on one type of heuristics for selections of samples in the beginning and later using a different one for *fine-tuning* M. (see also Section 4.3). This is in

Algorithm 1 Imitating Active Learner Ensembles

1: data $\mathcal{D}$, labeled validation data $\mathcal{D}_{\text{val}}$, classifier M, budget $\mathcal{B}$, experts $\mathcal{E}$, acquisition size acq, subset size n, probability p, states $\mathcal{S}$, actions $\mathcal{A}$, random policy π (acq ≥ 1, $n = 100$).
2: **for** e $= 1 \ldots$ episodes$_{\max}$ **do**
3: $\quad \mathcal{D}_{\text{lab}}, \mathcal{D}_{\text{pool}} \leftarrow \text{split}(\mathcal{D})$
4: $\quad$ **repeat**
5: $\quad\quad M \leftarrow \text{initAndTrain}(M, \mathcal{D}_{\text{lab}})$
6: $\quad\quad \mathcal{D}_{\text{sub}} \leftarrow \text{sample}(\mathcal{D}_{\text{pool}}, n)$
7: $\quad\quad e^* \leftarrow \text{bestExpert}(\mathcal{E}, M, \mathcal{D}_{\text{sub}}, \mathcal{D}_{\text{val}})$
8: $\quad\quad \mathcal{D}_{\text{sel}} \leftarrow e^*.\text{SelectQuery}(M, \mathcal{D}_{\text{sub}}, \text{acq})$
9: $\quad\quad \mathcal{S}, \mathcal{A} \leftarrow \text{toState}(M, \mathcal{D}_{\text{sub}}, \mathcal{D}_{\text{lab}}), \text{toAction}(\mathcal{D}_{\text{sel}})$
10: $\quad\quad$ **if** $\text{Rnd}(0, 1) \geq p$ **then**
11: $\quad\quad\quad$ // We may choose π's selection
12: $\quad\quad\quad \mathcal{D}_{\text{sel}} \leftarrow \pi.\text{SelectQuery}(M, \mathcal{D}_{\text{sub}}, \text{acq})$
13: $\quad\quad$ **end if**
14: $\quad\quad \mathcal{D}_{\text{lab}} \leftarrow \mathcal{D}_{\text{lab}} \cup \mathcal{D}_{\text{sel}}$
15: $\quad\quad \mathcal{D}_{\text{pool}} \leftarrow \mathcal{D}_{\text{pool}} \setminus \mathcal{D}_{\text{sel}}$
16: $\quad\quad$ Update policy using $\{\mathcal{S}, \mathcal{A}\}$
17: $\quad$ **until** $|\mathcal{D}_{\text{lab}}| > \mathcal{B}$
18: **end for**

line with previous research that combines uncertainty- and density-based heuristics and that learns an adaptive combination framework that weights them over the training course (Li and Guo, 2013). π learns a more suitable selection for the classifier's learning stage through introspection into the classifier's state. Note that this is more adaptive to new problems than e.g. encoding time directly (for instance as a function of the number of acquisitions).

3.3 Policy Training

Our policy training builds on the intuition behind DAgger, which is a well-known algorithm for IL that aims to train a policy by iteratively growing a dataset for supervised learning. The key idea is that the dataset includes the states that are likely to be visited over the course of solving a problem (in other words, those state and action encodings that would have been visited if we would follow a hard-coded AL strategy). To this end, it is common when using DAgger to determine a policy's next state by either *following* the current policy or an available expert (Ross et al., 2011). We thus grow a list of state and action pairs, and randomly either choose expert or policy selections as the action.

Each episode of the IL cycle lasts until the AL labeling budget is reached for $episodes_{max}$ iterations. We aggregate the *states* and *actions* over all episodes, and continually train the policy on the pairs. We use DAgger to further randomize the exploration of $\mathcal{D}$. Instead of always following the best expert's advice, we randomly follow the policy's prediction, and thus enrich the possible states.

Our IL approach for training π is given in Algorithm 1. At each AL cycle, we randomly sample a subset $\mathcal{D}_{\text{sub}}$ of n samples from the unlabeled pool $\mathcal{D}_{\text{pool}}$ (line 3). We find the best

expert e^* from a set of experts $\mathcal{E}$ (line 7) by extending the training dataset by the expert selections (from $\mathcal{D}_{\text{sub}}$) and train a classifier each. This means that each expert constructs one batch according to its heuristic, e.g., a batch composition could maximize model-change, and queries the oracle for labels. We choose the best expert by comparing the resulting classifiers' accuracies on the labeled validation dataset. We next set its acquisition as this iteration's chosen target and store *state* and *action* for the policy training (line 9). Depending on the probability p (line 10) we then either use the policy or the best expert to increase $\mathcal{D}_{\text{lab}}$ for the next iteration (line 14). After each episode we retrain π on the *state* and *action* pairs (line 16).

4. Experiments

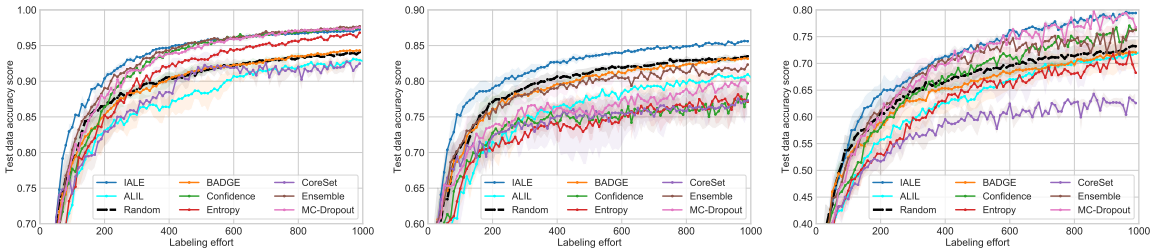
We first describe our experimental setup (Section 4.1). Next, we describe how we trained our policy on MNIST (Section 4.2) and evaluate our approach by transferring it to test datasets, i.e., to FMNIST and KMNIST (Section 4.3), and Extended MNIST, SVHN and CIFAR-10/-100 (Section 4.4). We end with a discussion of ablation studies and the limitations of our approach (Section 4.5). The source code is available at `https://github.com/crispchris/IALE` and can be used to reproduce our experimental results.

4.1 Experimental Setup

Datasets. We use the image classification datasets MNIST (LeCun et al., 1998), Fashion-MNIST (FMNIST) (Xiao et al., 2017), Kuzushiji-MNIST (KMNIST) (Clanuwat et al., 2018), Extended MNIST (Cohen et al., 2017), CIFAR-10/-100 (Krizhevsky, 2009), and SVHN (Netzer et al., 2011) for our evaluation. The MNIST-variants consist of $70,000$ grey-scale images (28×28px) in total for 10 classes. MNIST contains the handwritten digits $0-9$, FMNIST contains images of clothing (i.e., bags, shoes, etc.), and KMNIST consists of Hiragana characters. Extended MNIST contains, among others, a 26-class split of handwritten letters (28×28px) with 145,600 samples. SVHN, CIFAR-10 and CIFAR-100 are higher dimensional image classification datasets (32×32 pixels, 3 color channels) with 10, or 100 classes. The CIFAR-variants contain 60.000 images of objects and animals. SVHN contains 600.000 images of house numbers.

To evaluate `IALE` we train a policy π, run it on unseen datasets along with the baselines, and average the results (over 3 iterations). We denote the number of labeled samples in the experiments as *labeling effort* until a budget is reached, to be able to compare different acquisition sizes for the same total budget. The similarity between FMNIST and MNIST (that has previously been shown (Nalisnick et al., 2019)) and the difficulty of FMNIST (it has been shown to be a demanding dataset for AL methods (Hahn et al., 2019)) make these datasets a perfect combination to evaluate `IALE`. We show broader generalization on Extended MNIST and the higher dimensional SVHN and CIFAR datasets (5 repetitions). Appendix A.3.2 presents additional results for transferring π.

Architectures of classifier M. We use the same CNN architecture that has been employed in previous research (Gal and Ghahramani, 2016). Our model has two convolutional layers, followed by a max pooling and dense layer. We add dropout layers after the convolution and dense layers and use ReLU activations. A soft-max layer allows for classification. We



(a) CNN/MNIST (b) CNN/Fashion-MNIST (c) CNN/Kuzushiji-MNIST

Figure 3: Active learning performance of the trained policy (trained on MNIST), compared to the baseline approaches including *ALIL* (Liu et al., 2018), validated on MNIST and evaluated on FMNIST and KMNIST.

also provide results for using π on a simple MLP and on a more complex ResNet-18 (He et al., 2016), and supplemental results in Appendix A.3.1.

Architecture of policy π. Our policy model π uses an MLP with three dense layers with 128 neurons each. The first two dense layers are followed by a ReLU activation layer, whereas the final layer has only one neuron and the output of this layer is passed onto a sigmoid function to constrain the outputs to the range $[0, 1]$. and to further process it into an aggregating top-k operation.

Baselines. We compare our method with different well-known AL approaches from literature: *ALIL* (which we adapted from Liu et al. (2018) to work with image classification tasks), *MC-Dropout*, *Ensemble*, *CoreSet*, *BADGE*, *Confidence*-sampling, *Entropy*-sampling and a random sampling. Appendix A.1.1 provides a more details on the baselines.

Notation. We denote the unlabeled dataset as $\mathcal{D}_{\text{pool}}$, the already labeled data as $\mathcal{D}_{\text{lab}}$ and a labeled validation data $\mathcal{D}_{\text{val}}$. We randomly sub-sample $\mathcal{D}_{\text{sub}}$ of size n from $\mathcal{D}_{\text{pool}}$. We use a budget $\mathcal{B}$ and acquisition size acq to select $\mathcal{D}_{\text{sel}}$ from $\mathcal{D}_{\text{sub}}$. We derive the state $\mathcal{S}$ from a classifier M, e.g., a CNN or ResNet, to train `IALE`'s policy network π, i.e., an MLP with two hidden layers. In policy training, experts $\mathcal{E}$ propose actions $\mathcal{A}$. DAgger's hyper parameter p is the probability for following either the best expert or the policy π itself.

4.2 Policy Training and Validation

We use the `MNIST` dataset as our source dataset on which we train our policy for 100 episodes, with each episode containing data from an AL cycle. The initial amount of labeled training data is 20 samples (class-balanced). At each step of the active learning process, 10 samples are labeled and added to the training data until a labeling budget $\mathcal{B}$ of $1,000$ is reached. We use the AL heuristics *MC-Dropout*, *Ensemble*, *CoreSet*, *BADGE*, *Confidence* and *Entropy* as experts, and use $\mathcal{D}_{\text{val}}$ with 100 labeled samples to score the acquisitions of the experts. The pool dataset is sampled with $n = 100$ at each AL iteration. We choose $p = 0.5$ for means of comparison with the baselines (based on preliminary experiments, see Appendix A.2.1 on *Exploration-Exploitation*). We train the policy's MLP on the growing list of state and action pairs using the binary cross entropy loss from Equation 2 and use the Adam optimizer (Kingma and Ba, 2015) for 30 epochs with a learning rate of 10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, ϵ=10^{-8}, without any weight decay.

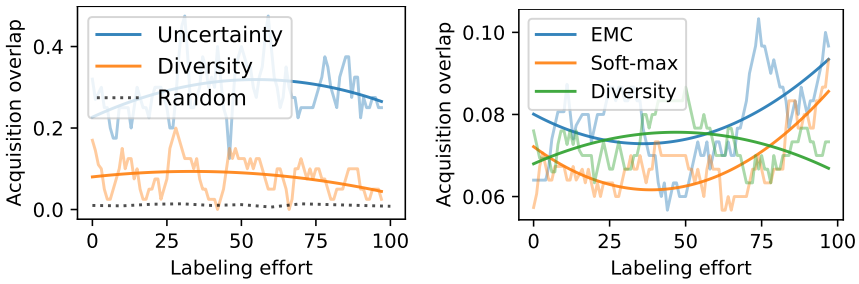
Figure 3a shows the results of our method in comparison to all the baseline approaches on MNIST, on which the policy was trained on. Our method consistently outperforms or is at least *en par* (towards the end, when enough representatives samples are labeled) with all the other methods. This finding on the policy-training dataset is not surprising, however, `IALE` performs better acquisitions than, e.g., *Ensemble* and *MC-Dropout*, for the important first half of the labeling budget, where it matters the most. In this experimental setting, *Confidence*-sampling performs similarly to the two more complex methods, even though it uses only the simple soft-max probabilities. While *Entropy* beats random sampling, it is still not competitive. *BADGE* performs similar to random sampling, which is due to the small acquisition size of 10 (the better performance of *BADGE* was reported with much larger acquisition sizes of 100 to 10,000 in Ash et al. (2020) as its mix of uncertainty and diversity heuristic benefits from these). The same applies to *CoreSet*, however, here it performs worst on average over all experiments. This finding is in line with previous research (Sinha et al., 2019; Hahn et al., 2019) and can be attributed to a weakness of the used p-norm distance metric regarding high-dimensional data, called the *distance concentration phenomenon*. The accuracy of *ALIL* on MNIST is similarly low as *CoreSet*. Moreover, *ALIL* is designed to add only one sample to the training data at a time (no batch-mode).

A general finding regarding computational efficiency in active learning is that `IALE` is faster than most baselines. While *MC-Dropout* requires 20 forward passes to decide which samples it acquires, and *Ensembles* $N = 5$ forward passes, one for each model, our approach requires only 2 inferences (for $\mathcal{D}_{\text{sub}}$ and $\mathcal{D}_{\text{lab}}$). The support for batch-mode (instead of selecting single samples) and using expert heuristics' batch acquisitions (instead of rolling out training of random samples from a small subset), accelerates the training of `IALE` compared to *ALIL* by several orders of magnitude (6 minutes versus 215 minutes per epoch on one NVidia Tesla V100 GPU). In a quantitative evaluation (with a labeling budget of 10.000 samples, an acquisition size of 10, training a ResNet-18 for 100 epochs, same GPU as before) the run time for `IALE` is 10:17:31 (hh:mm:ss) vs. 9:45:12 for random sampling. Compared to 49:15:23 for *Ensembles*, 14:23:18 for *MC-Dropout* and 11:58:47 for *BADGE*, this shows that `IALE` is faster. Only two baselines *Conf* (10:12:01) and *Entropy* (10:05:21) run faster, but they perform worse than `IALE` and even worse than random sampling.

4.3 Policy Transfer and Testing

To evaluate π's performance, we have to run it on a different dataset than the one that it has been trained on. Hence, we train π on the source dataset MNIST as in Section 4.2 and use it for the AL problem on FMNIST and KMNIST. We use an initial class-balanced labeled training dataset of 20 samples and add 10 samples per AL acquisition cycle until we reach a labeling budget of 1,000 samples. All the baselines are evaluated along with our method for comparison.

Figures 3b and 3c show the performance of `IALE` along with the baselines on FMNIST and KMNIST. Still, `IALE` consistently outperforms the baselines on both datasets. We can see that it learns a combined and improved policy that outperforms the individual experts consistently and (sometimes) even with large margins. On FMNIST `IALE` is the only method that actually beats a random sampling (similar findings have previously been reported by Hahn et al. (2019)). `IALE` is consistently $1 - 3\%$ better than random sampling



(a) Uncertain and diverse. (b) Types of uncertainty.

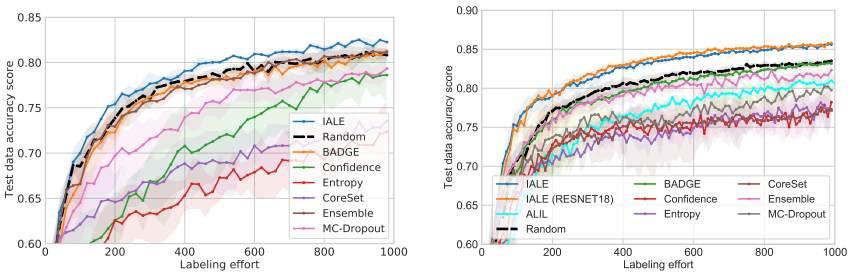
Figure 4: Complementary acquisition.

on FMNIST, on the harder KMNIST dataset `IALE` is even $7-9\%$ ahead. The baselines give a mixed picture. *ALIL*'s performance is not competitive on any task and actually never beats a random sampling strategy. We also see unstable performance for *MC-Dropout* and *Ensemble*, that generally perform similarly well. The simple soft-max heuristics *Entropy* and *Confidence* fail on FMNIST. *CoreSet* lags far behind, especially on KMNIST. *BADGE* always performs like random sampling, due to the aforementioned problematic acquisition size.

Sample composition of acquisition batches. We compare `IALE`'s chosen samples with the ones chosen by the experts, to gain insights on what `IALE` imitates and how the composition changes over the AL cycles. We evaluate all baseline experts and our method 5 times for 100 AL cycles on FMNIST ($|\mathcal{D}_{\text{sub}}| = 100$ and acq-size$= 10$), and report the results as well as fitted polynomials to highlight trends. In addition, we report the intersection of two i.i.d. randomly selected sets as the *Random* baseline with 1% overlap. For an acquisition size of 50, such a random overlap increases to 25%. Since the acquisitions between imitated baselines overlap, we are especially interested in their complementary acquisitions, e.g., samples that were only selected by a specific heuristic. Hence, we first separate AL into the families of *uncertainty*- and *diversity*-based methods. We group ensemble model combinations (EMCs) (Lakshminarayanan et al., 2017) (*Ensemble*, *MC-Dropout*) with single model soft-max methods (*Confidence*, *Entropy*), and compare with methods with diversity (*BADGE* and *CoreSet*). The results in Figure 4a show that the policy predominantly overlaps with uncertainty-based baselines. As Figure 4b shows, the exclusively by *EMCs* selected samples form the larger set. `IALE` may outperform any single heuristic due to its complementary strategy. The overlap between π and *any* other heuristic (intersection), decreases from about 80% to 60% over time. π selects samples that none of the experts choose, see Appendix A.2.2, where we also show π's overlap with single heuristics.

4.4 Policy Generalization

`IALE` learns a transferable AL policy. It unifies heuristics and generalizes over tasks and model architectures, because the state retains its formulation between tasks and architectures.



(a) MLP/Fashion-MNIST. (b) CNN/Fashion-MNIST.

Figure 5: (a) π (trained on CNN and MNIST) applied to MLP on FMNIST (b) π (trained on ResNet-18 and MNIST) applied to CNN on FMNIST.

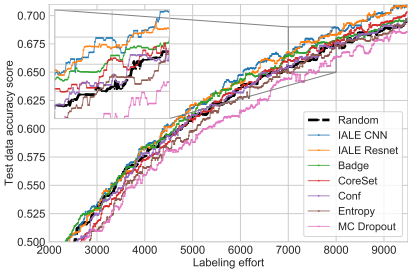
In this section, we evaluate the extend and limitations of the policy for transfers between MLP, CNN and ResNet classifiers, and on increasingly complex image datasets.[3]

Architecture transfer. Here, we explore the transfer of our approach to different architectures. We apply a policy, trained on CNN and MNIST, to an MLP (two hidden layers, 128 units, ReLU activation) on FMNIST and show the results in Figure 5a. Even with 5 random seeds and an increase batch size of 20 `IALE`, *BADGE* and *Ensemble* train classifiers stably, with `IALE` being on top. Next, we train a second policy on ResNet-18 and MNIST (`IALE ResNet`). We apply it to a CNN model on the FMNIST dataset and compare also with the previous policy (`IALE CNN` on MNIST). The results in Figure 5b show that both variants perform similarly despite their different policy training contexts. A potential explanation is that the policy learned similar decision strategies for both types (and sizes) of convolutional networks. This also shows that the state s and policy π are well-matched. First, the state s is rich enough for the policy π to learn and decode relevant information for generalizing the AL task. Second, the policy has a high enough capacity for transferring the policy to different networks and tasks, even though the embedding size itself is fixed.[4]

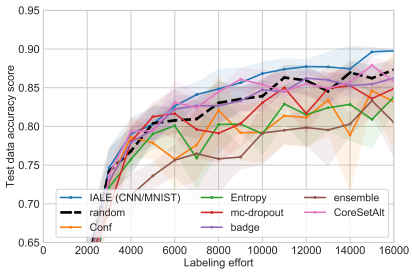
Both transfer experiments show `IALE`'s general ability to learn a *model-agnostic* AL strategy, of course within this experiment's scope. Experiments with deeper networks or an explanatory analysis of the policy's decision rules and state remain as future work.

Higher-dimensional data. Next, we evaluate `IALE` on the higher-dimensional CIFAR-10 and SVHN. For the latter, we increase the batch- or acquisition size to $1,000$ so that training converges. We re-use the two policies (CNN/MNIST and ResNet/MNIST) from the previous experiment, but train exclusively ResNet-18 classifiers, because the simpler classifier models (MLP, CNN) did not yield satisfying results for any AL strategy. In summary, we use $2,000$ initial labels, an acq-size of 10 and $\mathcal{B} = 10,000$ for CIFAR-10 and $1,000$ initial labels, an acq-size of $1,000$ and $\mathcal{B} = 16,000$ for SVHN. Figure 6a shows all results on CIFAR-10

3. The transfer of the two different π's works because the shape of the classifier's embedding is invariant to architecture and data.
4. While the size of the embedding of the training and test architectures need to match exactly, it is only one aspect of the proposed approach besides the full state and the policy's network. Specifically, given an embedding of sufficient size we can construct an adequate state for good generalization from that embedding and other information, i.e., predictive uncertainty and gradient information. Our approach does not only rely on the embedding and the constructed state. Instead, the policy network itself has a large capacity and decodes the state.

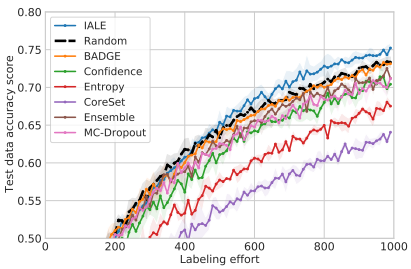


(a) ResNet/CIFAR-10.

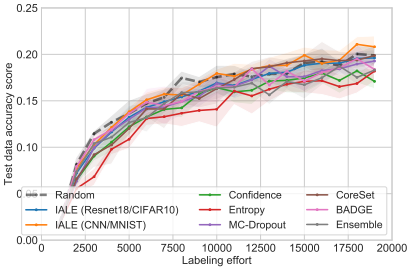


(b) ResNet/SVHN.

Figure 6: (a) π (trained on CNN (`IALE CNN`) or ResNet-18 (`IALE ResNet`) on MNIST) applied to ResNet-18 on CIFAR-10 (filtered). (b) π (trained on CNN and MNIST) applied to ResNet-18 on SVHN (filtered).



(a) CNN/Extended MNIST (letters).

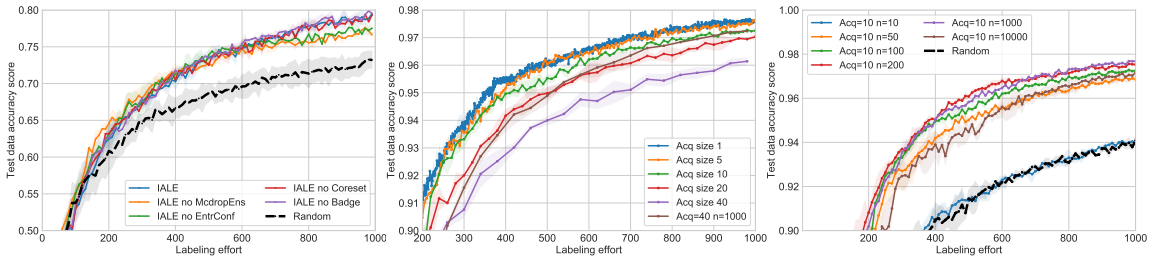


(b) ResNet/CIFAR-100.

Figure 7: (a) π (trained on CNN and MNIST) applied to CNN on Extended MNIST (letters). (b) similarly on CIFAR-100 (filtered).

with different policies: `IALE` is at least on par or better than a random sampling while the other experts are on par or worse than random sampling, some of them considerably. On SVHN, the performance gap opens wider with a larger batch-size (Figure 6b). Here, `IALE` performs best and is the only AL method that consistently beats a random sampling, with a relatively large margin. Furthermore, we want to emphasize that `IALE` generalizes beyond its budget of $1,000$ during policy training to longer time horizons of budgets like $10,000$ or even $16,000$. This is a benefit of our policy's introspective, state-based learning over alternatives like optimization-based meta-learning (Ravi and Larochelle, 2017; Chen et al., 2017), that struggles with longer time horizons as shown by Mishra et al. (2018) and Chen et al. (2017). Finally, these results show, that our framework learns a *task-agnostic* AL strategy for the presented image datasets. See Appendix A.3.2 for raw results.

Arbitrary class count. We perform experiments on Extended MNIST (26 classes; letters) and CIFAR-100 (100 classes) to increase the complexity of the classification task. To do this we must remove the fixed-length vectors *prediction* and *empirical class distribution* from the policy's state. Even though this may lead to a slightly poorer performance (see Appendix A.4.3) this allows transfers to tasks with an arbitrary numbers of classes. To start, we train two new policies with the reduced state, one with CNN on MNIST and another



(a) Experts. (b) Varying acquisition sizes. (c) Varying $|\mathcal{D}_{\text{sub}}|$.

Figure 8: Ablation studies on `IALE` for (a) different expert sets (on KMNIST), (b) acq-sizes and (c) n=$|\mathcal{D}_{\text{sub}}|$ (on MNIST).

with a ResNet-18 on CIFAR-10. Then, we apply `IALE` to a CNN on the Extended MNIST dataset.

Figure 7a shows the results from which we can draw two important observations: (1) `IALE` can be applied to problems with arbitrary class count, and (2) `IALE` still performs well compared to baselines. Next, we explore the limits by applying both policies to ResNet-18 on CIFAR-100 ($1,000$ initial labels, an acq-size of $1,000$ and $\mathcal{B} = 19,000$). Figure 7b shows that while `IALE` is still on par or better than the baselines, random sampling works surprisingly well, and shows the limitations of current AL heuristics, leaving space for future research.

4.5 Ablation Studies

Varying experts. To investigate the influence of experts, we leave out some types of experts: We categorize them into 4 simple groups, i.e., EMCs (*McdropEns*), soft-max uncertainty (*EntrConf*), diversity (*Coreset*) and hybrid (*Badge*), and leave one subset out. We fully train each method on MNIST with $\mathcal{B} = 1,000$ and an acquisition size of 10, and present the results of the evaluation on KMNIST in Figure 8a (more results including the ablation of state elements can be found in Appendixes A.4.2 and A.4.3). We see that most combinations perform well compared to the baselines. However, leaving out uncertainty-based heuristics can decrease performance, as they contribute the largest fraction to `IALE`'s selection composition (see Section 4.3). Even though training time is longer with *MC-Dropout*, the gains in performance can be worth it. In contrast, the soft-max uncertainty-based heuristics are computationally cheap and yield well-performing policies.

Hyperparameters. Two important parameters are the acquisition size *acq* and the size of $\mathcal{D}_{\text{sub}}$. Machine learning engineers may specifically be interested in modifying the acquisition size according to practical constraints. Hence, we show that arbitrary values are possible when applying π. Figures 8b and 8c show results for applying the policy for acq of 1 to 40 and size of $\mathcal{D}_{\text{sub}}$ between n=10 and n=$10,000$. During policy training we fixed acq= 10 and size of $\mathcal{D}_{\text{sub}}$ to n=100. This section also addresses the question of how `IALE` learns to sample diverse sets of points. Our empirical study shows that it does not sample non-diverse sets, which could be a failure state. Varying the acquisition size and $\mathcal{D}_{\text{sub}}$ produces the following insights. As expected, `IALE` performs best at acq= 1 and worst at acq= 40, if n is unchanged (because n limits the available choices), e.g., bad samples are chosen. Increasing n to $1,000$ alleviates this. However, there is an upper limit to the

size of $\mathcal{D}_{\text{sub}}$ after which performance deteriorates again, see Figure 8c. We believe that random sub-sampling simplifies the selection of diverse, uncertain samples. The performance decrease for small and large n supports this hypothesis. An optimal size of the sub-sampled data could be determined for different active learners, similarly to how some acquisition sizes are more suitable than others (see diversity vs. uncertainty in Appendix A.4). However, this additional interesting finding is not investigated further within this paper. The lower limit becomes apparent again when n is smaller than 10 times acq, with n =acq essentially being a random sampling. From our observations, n should be $10 - 100$ times acq (for 10 classes). From the small differences within this value range, it is suggested that our method is suitable for larger acquisition sizes for batch-mode AL, as its performance is not affected much.

5. Conclusion

We proposed a novel imitation learning approach for active learning. Our method learns to imitate the behavior of different active learners, such as uncertainty-, diversity-, model change- and query-by-committee-based heuristics, on one initial dataset and model, and transfers the obtained knowledge to work on other datasets and models (that share an embedding space). Our policy π is a simple MLP that learns a unified strategy from the experts based on a state with high capacity that contains gradient signals, embeddings and statistics of the data. Our experiments on different image datasets (four MNIST variants, CIFAR-10/100, SVHN) and model architectures (MLP, CNN, ResNet) show that `IALE` outperforms the state of the art and learned a complementary strategy. An ablation study and analysis of the influence of certain hyper-parameters also shows the limitations of our approach.

Future work investigates alternatives to the sampling step, as it may lead to sub-optimal choices from very large or imbalanced datasets. This would require a different loss than cross-entropy, in order to retain the ordering information, and could lead to a reformulation as a learning-to-rank problem of (compatible) experts' choices instead. Finally, an analysis of how π's *state* enables a transfer of its active learning strategy between classifier architectures and datasets may lead to some level of explanation of the principles of deep active learning.

Acknowledgments

We would like to acknowledge support for this project from the Bavarian Ministry of Economic Affairs, Infrastructure, Energy and Technology as part of the Bavarian project Leistungszentrum Elektroniksysteme (LZE) and the Center for Analytics-Data-Applications (ADA-Center) within the framework of "BAYERN DIGITAL II".

Appendix A.

In this section we provide an extension of the experiments section (Section 4) and feature additional results that support a more complete evaluation of `IALE`. We adhere to the same section structure.

A.1 Experimental Setup

A.1.1 Baselines

In the following is a short explanation of the baselines and experts that we used in our experiments:

1. *Random Sampling* randomly samples data points from the unlabeled pool.

2. *MC-Dropout* (Gal et al., 2017) approximates the sample uncertainty of the model by repeatedly computing inferences of the sample, i.e., 20 times, with dropout enabled in the classification model.

3. *Ensemble* (Beluch et al., 2018) trains an ensemble of 5 classifiers with different weight initializations. The uncertainty of the samples is quantified by the disagreement between the model predictions.

4. *CoreSet* (Sener and Savarese, 2018) solves the k-center problem using the pool-embeddings of the last dense layer (128 neurons) before the soft-max output to pick samples for labeling.

5. *BADGE* (Ash et al., 2020) uses the gradient of the loss (given pseudo labels), both its magnitude and direction, for k-means++ clustering, to select uncertain and diverse samples from a batch.

6. *Confidence*-sampling (Wang and Shang, 2014) selects samples with the lowest class probability of the soft-max predictions.

7. *Entropy*-sampling (Wang and Shang, 2014) calculates the soft-max class probabilities' entropy and then selects samples with the largest entropy, i.e., where the model is least certain.

8. *ALIL* (Liu et al., 2018): we modify *ALIL*'s implementation (that is initially intended for NLP tasks) to work on image classification task. Due to the high runtime costs of running *ALIL* (as the acquisition size is 1), we perform the training of *ALIL* for 20 episodes. We trained the *ALIL* policy network with a labeling budget $\mathcal{B}$ of $1,000$ and an up-scaled policy network comparable to that of our method along with a similar M as we use to evaluate the other AL approaches. We left the coin-toss parameter p at 0.5, and the k parameter for sequential selections from a random subset of $\mathcal{D}_{\text{pool}}$ at 10.

We use the variation ratio metric (Gal et al., 2017) to quantify and select the data samples for labeling from the uncertainty obtained from *MC-Dropout* and *Ensemble* heuristics. The

variation ratio metric is given by its Bayesian definition (Gal et al., 2017) for a data sample $x \in \mathcal{D}_{\text{pool}}$ in Equation 3 and for an ensemble expert (Beluch et al., 2018) in Equation 4:

$$\text{variation-ratio}(x) = 1 - max_y p(y|x, D) \tag{3}$$

$$= 1 - \frac{m}{N}, \tag{4}$$

where m is number of occurrences of the mode and N is the number of forward passes or number of models in the ensemble.

A.1.2 Datasets

We show samples of the three datasets MNIST, Fashion-MNIST and Kuzushiji-MNIST in Figure 9 to illustrate their similarity. Extended MNIST consists of handwritten letters, while the other image datasets used in the evaluation (CIFAR and SVHN) differ greatly and have color information.

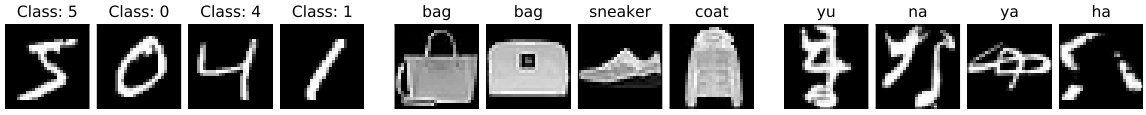
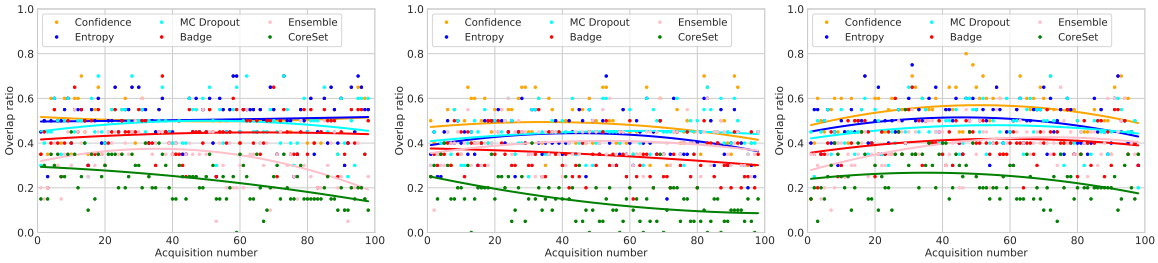


Figure 9: Examples for the three datasets MNIST, Fashion-MNIST, and Kuzushiji-MNIST.



(a) MNIST (b) Fashion-MNIST (c) Kuzushiji-MNIST

Figure 10: The overlap plots for all datasets MNIST, FMNIST and KMNIST datasets.

A.2 Policy Training

A.2.1 Exploration-Exploitation in DAgger

DAgger uses a hyper-parameter p that determines how likely π predicts the next action, and thereby setting the next state, instead of using the best expert from $\mathcal{E}$. In this preliminary study we compare the influence it has to either fix p to 0.5 or to use an exponential decay parameterized by the number of the current episode epi: $1 - 0.9^{epi}$. We train the policy on MNIST for 100 episodes with a labeling budget of 1,000 and an acquisition size of 10 (as before). Our result is that the *fixed* policy outperforms the *exponential* one by a small margin for the transfer of the policy to another dataset than the trained one, which is in line with previous findings (Liu et al., 2018).

A balanced (i.e., fixed) ratio does not emphasize one over the other, whereas an exponentially decay quickly relies on the policy for selecting new states of the dataset, and thus it trains on too few optimal states over the AL cycle.

A.2.2 Overlap ratios

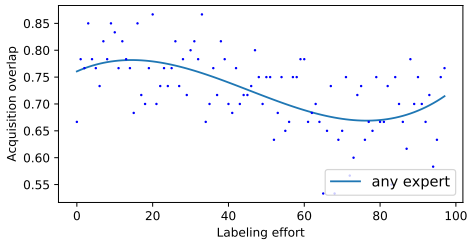


Figure 11: Overlap with any expert on Fashion-MNIST.

In addition to analyzing complementary compositions `IALE`'s acquisitions, we show overlaps with each baselines independently. This detailed view shows which heuristic π imitates the most. The overlap is given in percent in relation to the baselines (see Figure 10) for different datasets. We plot second-order polynomials, fit to the percentages (given as dots) over 100 acquisitions of size 10. Interestingly, the overall overlap is lower on FMNIST, where our method is the only one that beats a random sampling. We confirm again that π mostly imitates uncertainty-based heuristics, i.e., soft-max heuristics and *MC-Dropout*, and the uncertainty-/diversity-heuristic *BADGE* (close behind). *Ensemble* is overlapping mostly at the beginning. *CoreSet* has the lowest overlap. Interestingly, the policy chooses about half of the samples differently from any single baseline. On the other hand, the overlap with *any* expert is relatively high but decreases over the AL cycle (see Figure 11). In other words, the policy selects a portion of samples that none of the experts selected. Note that `IALE`'s acquisitions are build from combinations of the heuristics (instead of single votes), as we show in Section 4.3. Here, the percentages do not sum up to 1 as the overlap ratios between baseline and `IALE` are independent and may also overlap with each other.

A.3 Policy Transfer

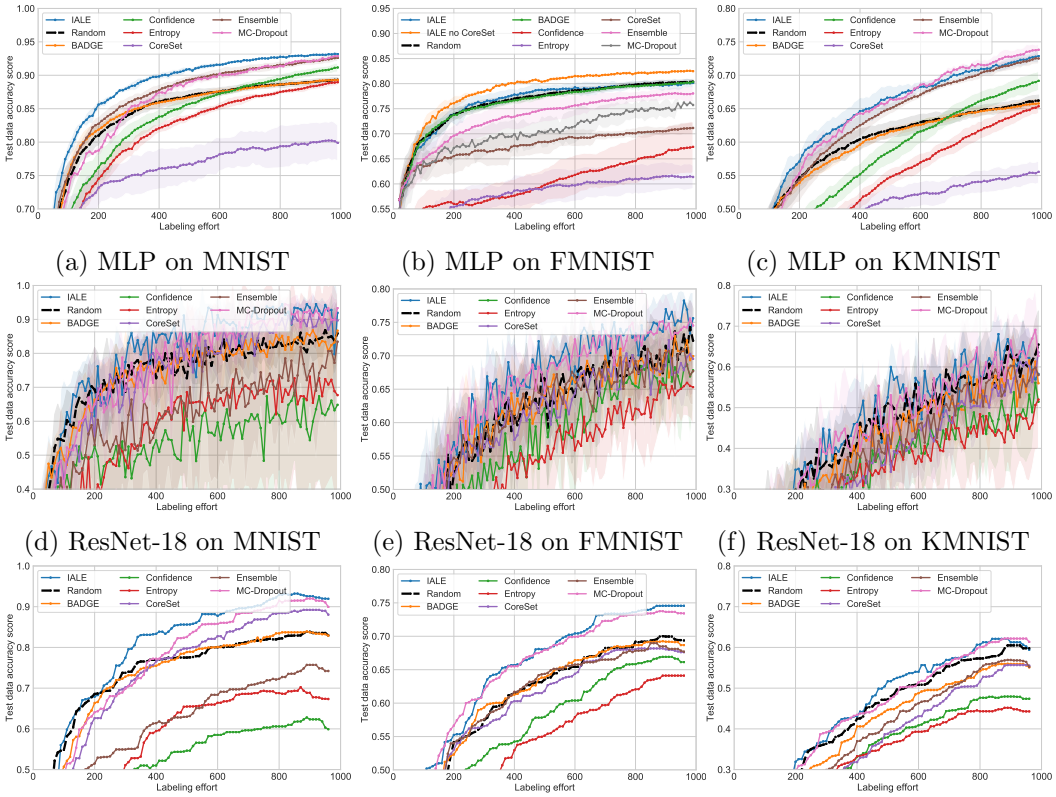
In this section we provide additional results on our studies on how our method performs in regard to applying it to unseen scenarios. These include that we use different datasets and classifier models in training and application of the policy. We show that π learns a task-agnostic AL strategy, that outperforms the baselines.

A.3.1 Classifier Architecture

In the evaluation Section 4, we show that our method is not bound to a specific classifier architecture. Here, we add results for the MLP architecture and give raw curves for a ResNet-18 classifier. To train `IALE`, we train policies on MNIST and apply them to all MNIST variants. All results for the experiments are given in Figure 12. We see the robustness of π over fundamentally different classifier architectures (2 to 18 layers). The deviations

for ResNet-18 are very large due to the very deep architecture and the modest amount of training data. We use median filtering in Figures 12g, 12h, and 12i.

These experiments show that π can learn AL strategies for both very small and very deep architectures and still outperform baselines. Even though the strongest baselines, i.e., *CoreSet* and *MC-Dropout*, come close to our method in accuracy, they are less versatile and require more computational resources, that is especially noticeable on deeper architectures.



(a) MLP on MNIST (b) MLP on FMNIST (c) MLP on KMNIST

(d) ResNet-18 on MNIST (e) ResNet-18 on FMNIST (f) ResNet-18 on KMNIST

(g) ResNet-18 on MNIST (h) ResNet-18 on FMNIST (i) ResNet-18 on KMNIST

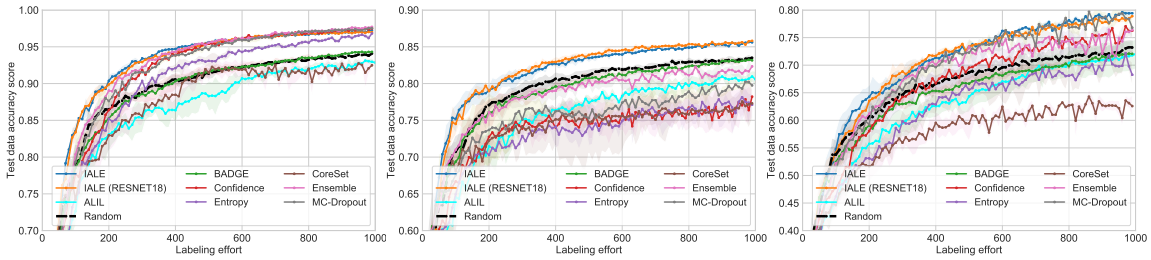
Figure 12: MLP and ResNet-18 classifiers, data averaged or median filtered. Active learning performance of the trained policy in comparison with the baseline approaches on MNIST, FMNIST and KMNIST datasets.

A.3.2 Classifier Architecture and Dataset

In Section 4.4, we show that π learns active learning independent from dataset and classifier. Here, we show additional results that mix both the source datasets *and* the classifiers.

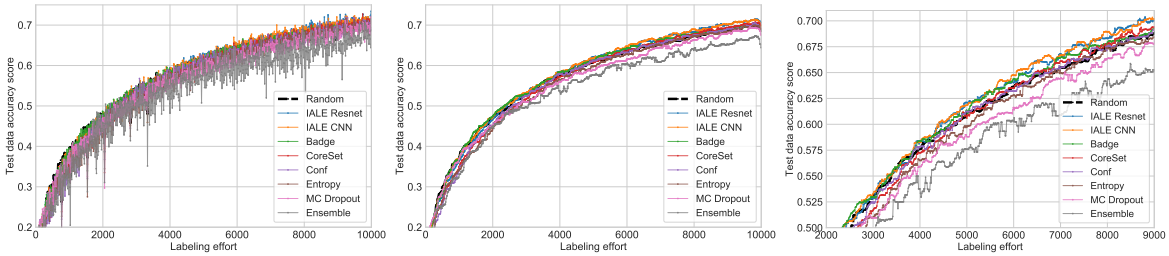
We report the results for applying π (trained on ResNet-18 and MNIST) to a CNN and all MNIST variants in Figure 13. `IALE` is always performing at the top, showing that it learns a *model- and task-agnostic* active learning strategy that transfers well.

CIFAR-10. We show additional results for applying π to a ResNet-18 classifier on `CIFAR-10`. To reiterate, we use two different π: π_1 was trained using ResNet-18 and MNIST (`IALE ResNet`) and π_2 was trained using CNN and MNIST (`IALE CNN`). The complete results



(a) MNIST (b) Fashion-MNIST (c) Kuzushiji-MNIST

Figure 13: Applying a policy trained using a ResNet-18 classifier (trained on MNIST) to a CNN-based classifier (on MNIST, FMNIST and KMNIST).



(a) full (raw) (b) full (filtered) (c) detail (filtered)

Figure 14: Full and enlarged segments of learning curve: Applying π trained on a CNN (IALE CNN) or ResNet-18 (IALE ResNet), trained on MNIST, to a ResNet-18 classifier on CIFAR10.

in Figure 14 are noisy due to the acquisition size of 10, and we report the raw learning curves (Figure 14a) and median filtered learning curves (Figure 14b). The most interesting segment of the learning curve is in Figures 14c in more detail and filtered. The results generally show the feasibility of transferring π to both different classifiers and datasets. `IALE` is on par or better than random sampling, and the other baselines are either on par or worse than random sampling (some of them considerably).

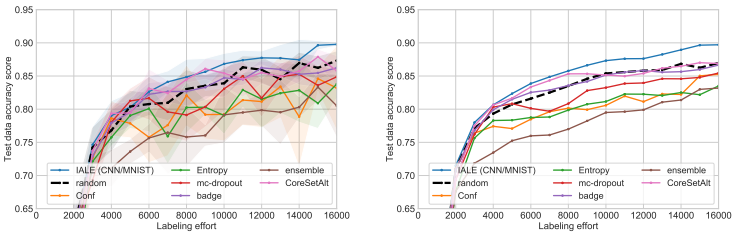
SVHN. We show the averaged learning curves for `SVHN` in Fig. 15a besides the smoothed averages for improved visibility in Fig. 15b. While the results exhibit some variance, we can clearly see that `IALE` performs best (and is the only AL methods that is consistently able to beat a random sampling.

While more experiments are certainly required to further emphasize these initial claims of generalizability to more diverse tasks, these findings are already very promising.

A.4 Ablation Studies

A.4.1 Hyperparameters.

We report fine-granular steps of acquisition sizes (see Figure 16a) with values between 1 and 10, plus 20 and 40, for $|\mathcal{D}_{\text{sub}}|$ of 100. Overall, a clear difference is not visible below 10 samples. For enhanced readability, we show a magnified section of the varied acquisition

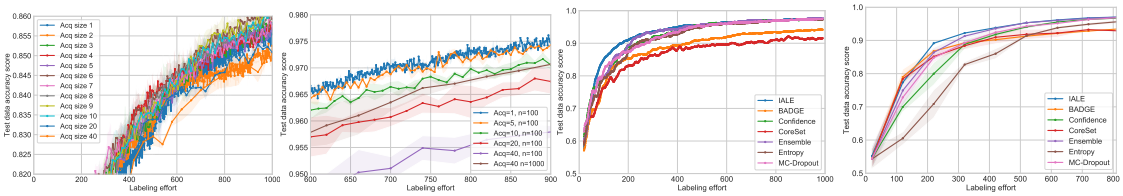


(a) SVHN. (b) SVHN.

Figure 15: The more complex dataset SVHN requiress more samples than MNIST variants. Learning curves as (a) averages and (b) smoothed plots).

sizes and $|\mathcal{D}_{\text{sub}}|$ in Figure 16b, that clearly shows the benefits of tuning $|\mathcal{D}_{\text{sub}}|$ to a suitable value for the acquisition size.

Acquisition sizes including baselines: We additionally compare the baseline active learning methods with our approach, as these exhibit different performance at different acquisition sizes, see Figure 16. We have included comparisons with acquisition sizes of either 1 or 100 (1 or 3 repetitions). For our method, for an acquisition size of 1 we chose $|\mathcal{D}_{\text{sub}}| = 100$ and for acquisition size of 100 we chose $|\mathcal{D}_{\text{sub}}| = 2,000$. While the results show that `IALE` outperforms the baselines they also highlight the large effect that the acquisition size has on some of the baseline methods. For instance, *CoreSet* constructs better set covers with larger batches, and *BADGE* increases its accuracy by constructing a representative sampling as well. At the same time the uncertainty-based methods, apart from *Entropy*, remain unaffected.

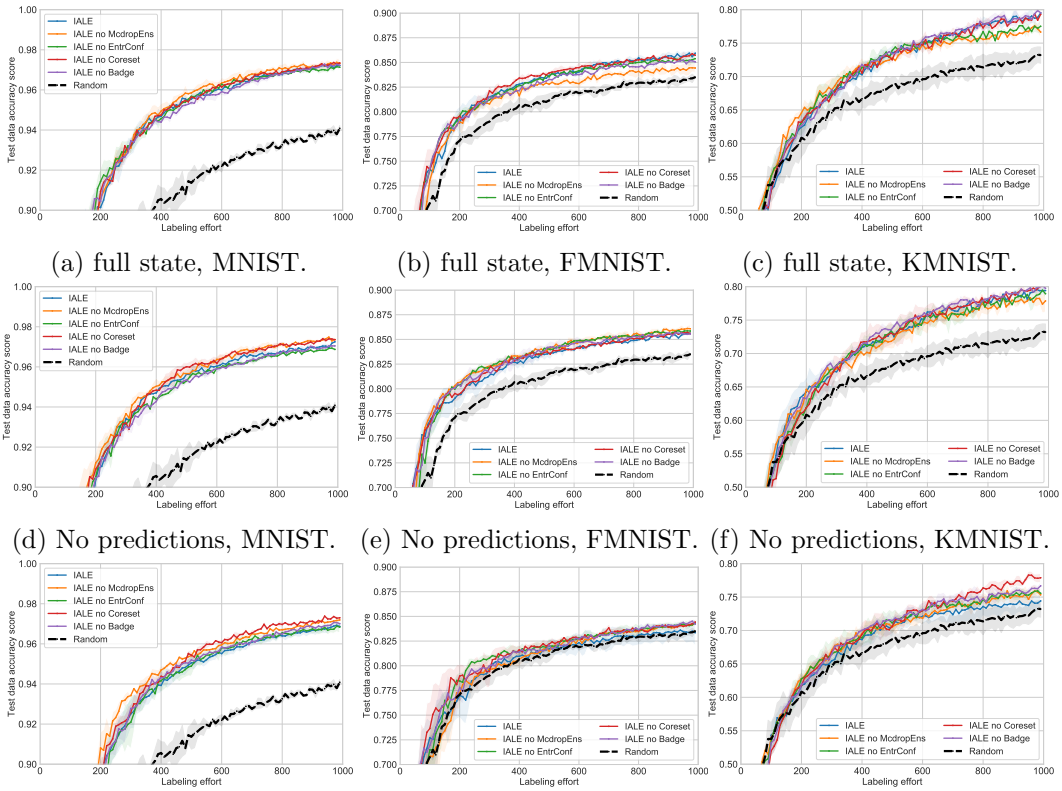


(a) Varying acq-sizes. (b) Varying acq-sizes (c) acq-size 1. (d) acq-size 100.

Figure 16: (a) Evaluating the acquisition sizes from 1 to 10 on FMNIST, and (b) varying different sub-pool sizes on MNIST. Diversity vs. uncertainty: Some experts are more suitable to other acquisition sizes, see (c) and (d), both evaluated on MNIST.

A.4.2 Varying Experts

We present more results for variations of sets of experts in Figure 17, and train the policies with the unchanged hyper-parameters and the CNN classifier on MNIST. The results for all three datasets show that the generally high performance of `IALE` holds for the leave-one-out sets of experts, with the full set of experts being consistently among the best performing policies.



(a) full state, MNIST. (b) full state, FMNIST. (c) full state, KMNIST.

(d) No predictions, MNIST. (e) No predictions, FMNIST. (f) No predictions, KMNIST.

(g) No gradients, MNIST. (h) No gradients, FMNIST. (i) No gradients, KMNIST.

Figure 17: The active learning performance for each (leave-one-out) set of experts. For partial state (without predictions, without gradients), we plot active learning performance for each (leave-one-out) set of experts.

A.4.3 Varying State Elements

Next, we study the state more closely. For unlabeled samples, the state contains two types of representations for predictive uncertainty: the statistics on predicted labels $M(x_n)$ and the gradient representations $g(M_e(x_n))$. In this study, we focus on leaving out one or the other. To get the full picture, we again train sets of experts for reduced states.

In Figure 17 we see that dropping gradients generally decreases performance (bottom row), while dropping predicted labels $M(x_n)$ affects performance very little (top row). However, the influence of different sets of experts is more important. We cannot see that a particular set of states and experts generally outperforms others consistently (while the negative effect of leaving out $g(M_e(x_n))$ is consistently visible). Overall, we find that using as many experts as available, combined with a full state both performs well and works reliably. Even though training a policy this way does not guarantee the best performance, it always performs among with the group of best policies.

References

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations (ICLR)*, Virtual Conference, Formerly Addis Ababa Ethiopia, 2020.

Philip Bachman, Alessandro Sordoni, and Adam Trischler. Learning algorithms for active learning. In *34th International Conference on Machine Learning (ICML)*, pages 301–310, Sydney, Australia, 2017.

Yoram Baram, Ran El-Yaniv, and Kobi Luz. Online choice of active learning algorithms. *Journal of Machine Learning Research*, (5):255–291, 2004.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2019 conference on machine translation. In *4th Conference on Machine Translation*, pages 1–61, Florence, Italy, 2019.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9368–9377, Salt Lake City, UT, 2018.

Arantxa Casanova, Pedro O. Pinheiro, Negar Rostamzadeh, and Christopher J. Pal. Reinforced active learning for image segmentation. In *International Conference on Learning Representations (ICLR)*, Virtual Conference, Formerly Addis Ababa Ethiopia, 2020.

Yutian Chen, Matthew W. Hoffman, Sergio Gómez Colmenarejo, Misha Denil, Timothy P. Lillicrap, Matt Botvinick, and Nando De Freitas. Learning to learn without gradient descent by gradient descent. In *International Conference on Machine Learning (ICML)*, volume 2, pages 1252–1260, Sydney, Australia, 2017.

Hong-Min Chu and Hsuan-Tien Lin. Can active learning experience be transferred? In *International Conference on Data Mining (ICDM)*, pages 841–846, Barcelona, Spain, 2016.

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. In *NeurIPS Workshop on Machine Learning for Creativity and Design*, Montreal, Canada, 2018.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926, Anchorage, AK, 2017.

Gabriella Contardo, Ludovic Denoyer, and Thierry Artières. A Meta-Learning Approach to One-Step Active-Learning. In *International Workshop on Automatic Selection, Configuration and Composition of Machine Learning Algorithms*, pages 28–40, Skopje, Macedonia, 2017.

Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. Learning to teach. *arXiv preprint arXiv:1805.03643*, 2018.

Meng Fang, Yuan Li, and Trevor Cohn. Learning how to active learn: A deep reinforcement learning approach. In *Conference on Empirical Methods in Natural Language Processing*, pages 595–605, Copenhagen, Denmark, 2017.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. *arXiv preprint arXiv:1806.02817*, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *33rd International Conference on Machine Learning (ICML)*, New York, NY, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *34th International Conference on Machine Learning (ICML)*, Sydney, Australia, 2017.

Lukas Hahn, Lutz Roese-Koerner, Peet Cremer, Urs Zimmermann, Ori Maoz, and Anton Kummert. On the robustness of active learning. In *5th Global Conference on Artificial Intelligence*, volume 65 of *EPiC Series in Computing*, pages 152–162, Bolzano, Italy, 2019.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, 2016.

Sepp Hochreiter, A. Steven Younger, and Peter R. Conwell. Learning to learn using gradient descent. In Georg Dorffner, Horst Bischof, and Kurt Hornik, editors, *Artificial Neural Networks — ICANN 2001*, pages 87–94, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

HM Sajjad Hossain, MD Abdullah Al Haiz Khan, and Nirmalya Roy. Deactive: Scaling activity recognition with active deep learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2:66:1–66:23, 2018.

Wei-Ning Hsu and Hsuan-Tien Lin. Active learning by learning. In *29th AAAI Conference on Artificial Intelligence (AAAI)*, page 2659–2665, Austin, Texas, 2015.

Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11719–11727, Long Beach, CA, 2019.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5574–5584, Long Beach, CA, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Learning active learning from data. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4225–4235, Long Beach, CA, 2017.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Discovering general-purpose active learning strategies. *arXiv preprint arXiv:1810.04114*, 2018.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6402–6413, Long Beach, CA, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Minghan Li, Xialei Liu, Joost van de Weijer, and Bogdan Raducanu. Learning to rank for active learning: A listwise approach. In *International Conference on Pattern Recognition (ICPR)*, Milano, Italy, 2020.

Xin Li and Yuhong Guo. Adaptive active learning for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Portland, OR, 2013.

Ming Liu, Wray Buntine, and Gholamreza Haffari. Learning how to actively learn: A deep imitation learning approach. In *56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1874–1883, 2018.

Christoffer Löffler, Christian Nickel, Christopher Sobel, Daniel Dzibel, Jonathan Braat, Benjamin Gruhler, Philipp Woller, Nicolas Witt, and Christopher Mutschler. Automated quality assurance for hand-held tools via embedded classification and automl. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, Ghent, Belgium, 2020.

Dwarikanath Mahapatra, Behzad Bozorgtabar, Jean-Philippe Thiran, and Mauricio Reyes. Efficient active learning for image classification and segmentation using a sample selection and conditional generative adversarial network. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 580–588, Granada, Spain, 2018.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

N Mishra, M Rohaninejad, X Chen, and P Abbeel. A Simple Neural Attentive Meta-Learner. In *International Conference on Learning Representations (ICLR)*, pages 1–17, Vancouver, Canada, 2018.

Andriy Mnih and Danilo J. Rezende. Variational inference for monte carlo objectives. In *33rd International Conference on Machine Learning (ICML)*, page 2188–2196, New York, NY, 2016.

Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations (ICLR)*, New Orleans, LA, 2019.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, Granada, Spain, 2011.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, pages 1–11, Toulon, France, 2017.

Sachin Ravi and Hugo Larochelle. Meta-learning for batch mode active learning. In *International Conference on Learning Representations (ICLR), Workshop Track Proc.*, Vancouver, Canaday, 2018.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 627–635, Ft. Lauderdale, FL, 2011.

Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Machine Learning: ECML 2006*, pages 413–424, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1289–1296, Vancouver, Canada, 2008.

Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *International Conference on Computer Vision (ICCV)*, pages 5972–5981, Seoul, South Korea, 2019.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4077–4087, Long Beach, CA, 2017.

LMA Tonnaer. Active learning in vae latent space. *Eindhoven University of Technology*, 2017.

Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Comp. Intelligence and Neuroscience*, 2018.

D. Wang and Y. Shang. A new active labeling method for deep learning. In *International Joint Conference on Neural Networks (IJCNN)*, pages 112–119, Beijing, China, 2014.

Mark Woodward and Chelsea Finn. Active one-shot learning. In *NeurIPS Deep Reinforcement Learning Workshop*, Barcelona, Spain, 2016.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.