SoCal: Selective Oracle Questioning for Consistency-based Active Learning of Cardiac Signals

Dani Kiyasseh
Department of Engineering Science
University of Oxford
Oxford, UK
dani.kiyasseh@eng.ox.ac.uk

Tingting Zhu
Department of Engineering Science
University of Oxford
Oxford, UK
tingting.zhu@eng.ox.ac.uk

David A. Clifton
Department of Engineering Science
University of Oxford
Oxford, UK
david.clifton@eng.ox.ac.uk

Abstract

The ubiquity and rate of collection of cardiac signals produce large, unlabelled datasets. Active learning (AL) can exploit such datasets by incorporating human annotators (oracles) to improve generalization performance. However, the over-reliance of existing algorithms on oracles continues to burden physicians. To minimize this burden, we propose SoCal, a consistency-based AL framework that dynamically determines whether to request a label from an oracle or to generate a pseudo-label instead. We show that our framework decreases the labelling burden while maintaining strong performance, even in the presence of a noisy oracle.

1 Introduction

The success of modern-day deep learning algorithms in the medical domain has been contingent upon the availability of large, labelled datasets (Poplin et al., 2018; Tomašev et al., 2019; Attia et al., 2019). Curating such datasets, however, is a challenge due to the time-consuming nature of, and high costs associated with, labelling. This is particularly the case in the medical domain where the input of expert medical professionals is required. One way of overcoming this challenge and exploiting large, *unlabelled* datasets is via the active learning (AL) framework (Settles, 2009). This framework iterates over three main steps: 1) a learner is tasked with acquiring unlabelled instances, usually through an acquisition function, 2) an oracle (e.g. physician) is tasked with labelling such acquired instances, and 3) the learner is trained on the existing and newly-labelled instances.

By altering the way in which acquisitions are performed and the degree of involvement of the oracle, the active learning framework aims to improve the performance of a network while minimizing the burden of labelling on the oracle. One principal desideratum for an acquisition function is its ability to reduce the size of the version space, the set of hypotheses (decision boundaries) consistent with the labelled training instances. This ability is highly dependent upon the approximation of the version space, a goal that Monte Carlo Dropout (MCD) attempts to achieve (see Fig. 1a). For example, state-of-the-art uncertainty-based acquisition functions, such as BALD (Houlsby et al., 2011), used alongside MCD acquire instances that lie in a region of uncertainty, a region where there is high disagreement between the hypotheses about a particular instance. In many scenarios, however, estimating this region of uncertainty is nontrivial. Furthermore, existing AL frameworks are overly *reliant* on the presence of an oracle. Such over-reliance precludes the applicability of AL algorithms to certain environments, such as low-resource healthcare settings, where an oracle is either unavailable or ill-trained for the task at hand.

Contributions. In this work, we design an active learning algorithm that better estimates the region of uncertainty and decreases its reliance on an oracle. Our contributions are as follows:

1. **Consistency-based active learning framework -** we propose a novel framework that stochastically perturbs inputs, network parameters, or both to guide the acquisition of unlabelled instances.
2. **Selective oracle questioning -** we propose a dynamic strategy which learns, for an acquired unlabelled instance, whether to request a label from an oracle or to generate a pseudo-label instead.

2 Related Work

Active learning methdologies were recently reviewed by Settles (2009). In the healthcare domain, Gong et al. (2019) propose to acquire instances from an electronic health record (EHR) database using a Bayesian deep latent Gaussian model to improve mortality prediction. Smailagic et al. (2018; 2019) acquire unannotated medical images by measuring their distance in a latent space to images in the training set. The work of Wang et al. (2019) is similar to ours in that they focus on the electrocardiogram (ECG). Gal et al. (2017) adopt BALD (Houlsby et al., 2011) in the context of Monte Carlo Dropout to acquire datapoints that maximize the Jensen-Shannon divergence (JSD) across MC samples. Previous work attempts to learn from multiple or imperfect oracles (Dekel et al., 2012; Zhang & Chaudhuri, 2015; Sinha et al., 2019). For example, Urner et al. (2012) propose choosing the oracle that should label a particular instance. Unlike our approach, they do not explore independence from an oracle. Yan et al. (2016) consider oracle abstention in an AL setting. Instead, we place the decision of abstention under the control of the learner. To the best of our knowledge, previous work, in contrast to ours, has assumed the existence of an oracle and has not explored a dynamic oracle selection strategy.

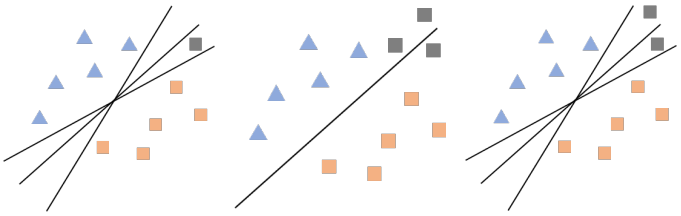
Consistency training in the context of semi-supervised learning helps enforce the smoothness assumption (Zhu, 2005). For example, Interpolation Consistency Training (Verma et al., 2019) penalizes networks for not generating a linear combination of outputs in response to a linear combination of inputs. Similarly, Xie et al. (2019) penalizes networks for generating drastically different outputs in response to perturbed instances. In the process, networks learn perturbation-invariant representations. McCallumzy & Nigamy (1998) introduce an acquisition function that calculates the average Kullback-Leibler divergence, $\mathcal{D}_{KL}$, between the output of a network and the consensus output across all networks in an ensemble. Unlike ours, their approach does not exploit perturbations. Similar to our work is that of Gao et al. (2019) which incorporates into the objective function a consistency-loss based on the $\mathcal{D}_{KL}$ and actively acquires instances using the variance of the probability assigned to each class by the network in response to perturbed versions of the same instance.

Selective classification imbues a network with the ability to abstain from making predictions. Chow (1970); El-Yaniv & Wiener (2010) introduce the risk-coverage trade-off whereby the empirical risk of a model is inversely related to its rate of abstentions. Wiener & El-Yaniv (2011) use a support vector machine (SVM) to rank and reject instances based on the degree of disagreement between hypotheses. In some frameworks, these are the same instances that active learning views as most informative. Cortes et al. (2016) outline an objective function that penalizes abstentions that are inappropriate and frequent. Most recently, Liu et al. (2019) propose the gambler's loss to learn a selection function that determines whether instances are rejected. However, this approach is not implemented in the context of AL. Most similar to our work is SelectiveNet (Geifman & El-Yaniv, 2019) where a multi-head architecture is used alongside an empirical selective risk objective function and a percentile threshold. However, their work assumes the presence of ground-truth labels and, therefore, does not extend to unlabelled instances.

3 Background

3.1 Active Learning

Consider a learner $f_\omega : x \in \mathbb{R}^m \rightarrow v \in \mathbb{R}^d$, parameterized by ω, that maps an m-dimensional input, x, to a d-dimensional representation, v. Further consider $g_\phi : v \in \mathbb{R}^d \rightarrow y \in \mathbb{R}^C$ that maps a d-



(a) MCD (b) MCP (c) BALC

Figure 1: Labelled instances from two classes and unlabelled instances (gray) alongside the Version space of (a) MCD where each MC sample is viewed as a distinct hypothesis (decision boundary), (b) MCP where there is one hypothesis but several perturbations of the unlabelled instance, and (c) BALC where there are several hypotheses in addition to the unlabelled instance and its perturbed counterpart.

dimensional representation, v, to a C-dimensional output, y, where C is the number of classes. After training on a pool of labelled data $L = (X_L, Y_L)$ for τ epochs, the learner is tasked with querying the unlabelled pool of data $U = (X_U, Y_U)$ and acquiring the top b fraction of instances, $x_b \sim X_U$, that it deems to be most informative. The degree of informativeness of an instance is determined by an acquisition function, α, such as BALD (Houlsby et al., 2011). Additional acquisition functions can be found in Appendix A. These are typically used in conjunction with Monte Carlo Dropout (Gal & Ghahramani, 2016) to identify instances that lie in the region of uncertainty, a region in which hypotheses disagree the most about instances.

4 Methods

4.1 Consistency-based Active Learning

4.1.1 Monte Carlo Perturbations

Unlabelled instances in proximity to the decision boundary are likely to be more informative for training than those further away. To identify such instances, we stochastically perturb them and observe the network's outputs. The intuition is that such outputs will differ significantly across the perturbations for instances close to the decision boundary (see Fig. 1b). We refer to this setup as Monte Carlo Perturbations (MCP) and illustrate its derivation in Appendix B.

4.1.2 Bayesian Active Learning by Consistency

Acquisition functions dependent upon perturbations applied to either the inputs (MCP) or the network parameters (MCD) alone can fail to identify instances that lie in the region of uncertainty. We illustrate this point with the following example: without loss of generality, let us assume an unlabelled instance is in proximity to some decision boundary A and is classified by the network as belonging to some arbitrary class 3. Such proximity should deem the instance informative for the training process (Settles, 2009). In the MCD setting, perturbations are applied to parameters, generating various decision boundaries, which in turn influence the network outputs. In Fig. 2 (red rectangle), we visualize such outputs for three arbitrary classes. If these parameter perturbations happen to be too small in magnitude, for example, then the network will continue to classify the instance as belonging to the same class. At this stage, regardless of whether an uncertainty-based or a consistency-based acquisition function is used, the instance would be deemed *uninformative*, and thus not acquired. As a result, an instance that should have been acquired (due to its proximity to the decision boundary) was *erroneously* deemed uninformative. A similar argument can be extended to MCP.

By applying perturbations to both instances and network parameters, we aim to leverage the smoothness assumption (Zhu, 2005) to better identify instances that lie in the region of uncertainty and thus avoid missing their acquisition. Motivated by this, we propose a framework, entitled Bayesian Active Learning by Consistency (BALC) (see Fig. 1c), that consists of three main steps: 1) we perturb an instance, x, to generate z, 2) we perturb the network parameters, ω, to generate ω', and 3) we pass

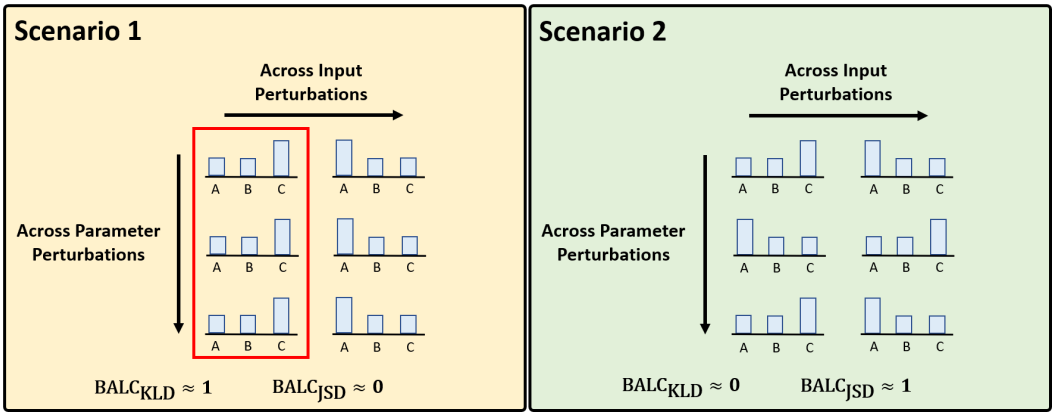


Figure 2: **(Scenario 1)** network output variations (for three classes, A, B, and C) caused primarily by input perturbations. The red rectangle illustrates a potential limitation of MCD. Given that these parameter perturbations do not result in network output variations and MCD is dependent upon said perturbations, unlabelled instances can be erroneously deemed uninformative. **(Scenario 2)** network output variations caused by both input and parameter perturbations. We show that while BALC_{KLD} is likely to acquire instances due to input perturbations, BALC_{JSD} considers both input and parameter perturbations when performing acquisitions.

both instances through the *perturbed* network, generating outputs, $p(y|x, \omega')$ and $p(y|z, \omega') \in \mathbb{R}^C$, respectively. We perform these steps for T stochastic perturbations and generate two matrices of network outputs, $G(x), G'(z) \in \mathbb{R}^{T \times C}$. We visualize such network outputs in Fig. 2 where $T = 3$ and $C = 3$.

To leverage G and G', we propose two divergence-based acquisition functions that acquire instances that the network is least robust to. In BALC_{KLD}, we calculate the D_{KL} between two C-dimensional Gaussians that are empirically fit to G and G'.

$$\text{BALC}_{\text{KLD}} = \mathcal{D}_{KL}(\mathcal{N}(\mu(x), \Sigma(x)) \parallel \mathcal{N}(\mu(z), \Sigma(z))) \tag{1}$$

where $\mu = \frac{1}{T}\sum_i^T G$ and $\Sigma = (G - \mu)^T(G - \mu)$ represent the empirical mean vector and covariance matrix of the network outputs, respectively. BALC_{KLD} is likely to detect output variations due to input perturbations. We support this claim in Fig. 2 by illustrating two scenarios. In scenario 1, network output variations are caused solely by input perturbations. In contrast, in scenario 2, network output variations are caused by both input and parameter perturbations. We show that $\text{BALC}_{\text{KLD}} \approx 1$ and 0 in these two scenarios, respectively. Since the higher the value of an acquisition function, the more informative an instance is, these scenarios illustrate BALC_{KLD}'s preference for input perturbations. To detect variations due to both input *and* parameter perturbations, we introduce BALC_{JSD} whose full derivation can be found in Appendix C.

$$\text{BALC}_{\text{JSD}} = \overbrace{\mathbb{E}_{i \in T}\left[\mathcal{D}_{KL}(G_i(x) \parallel G'_i(z))\right]}^{\text{across parameter perturbations}} - \overbrace{\mathcal{D}_{KL}(\mathbb{E}_{i \in T}[G(x)] \parallel \mathbb{E}_{i \in T}[G'(z)])}^{\text{across input perturbations}} \tag{2}$$

4.2 Tracked Acquisition Function

Deriving the informativeness of unlabelled instances based solely on the value of the acquisition function at a *single* epoch can be erroneous. This is partially driven by limitations in the approximation of the version space, which is known to hinder performance (Cohn et al., 1994). To improve this approximation, we propose to *track* an acquisition function over time (e.g., epochs) before employing it to acquire instances. The intuition is that by incorporating temporal information, we accumulate hypotheses in the version space and thus obtain a more reliable estimate of the relative informativeness of each instance. Acquiring such instances would help reduce the size of the version space at a greater rate. For any tracked acquisition function, $\alpha(t)$, the corresponding area under the temporal acquisition function, AUTAF $\in \mathbb{R}^1$, is calculated as follows:

$$\text{AUTAF} = \int_0^\tau \alpha(t)dt \approx \sum_{t=0}^{\tau}\left(\frac{\alpha(t + \Delta t) + \alpha(t)}{2}\right)\Delta t \tag{3}$$

where the integral is approximated using the trapezoidal rule, Δt is the time-step (in epochs) between epochs at which ordinary acquisition values are calculated, and τ is the epoch at which the AUTAF is calculated and an acquisition of unlabelled instances is performed.

4.3 Selective Oracle Questioning

Alongside our consistency-based AL framework, we aim to minimize the burden of labelling on an oracle. To do so, we learn a network that dynamically determines whether to request a label from an oracle or to generate a pseudo-label for each acquired unlabelled instance. We refer to this strategy as selective oracle questioning in active learning (SoQal).

Oracle selection network. In addition to the learners outlined in Sec. 3.1, f_ω and g_ϕ, we introduce an oracle selection network, $h_\theta : v \in \mathbb{R}^d \rightarrow t \in [0, 1]$ parameterized by θ, that maps d-dimensional representations, v, to a scalar, t, as shown in Fig. 3.

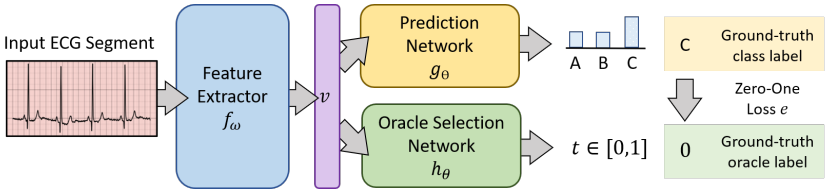


Figure 3: An instance, x, is provided as input to the feature extractor, f_ω. The representation, v, is passed through g_ϕ to generate task predictions, and through h_θ to help determine whether a label is requested from an oracle. The zero-one loss of g_ϕ acts as the ground-truth label for t where $t \approx 1$ indicates a hard-to-classify instance that would benefit from an oracle label. Otherwise, the instance is pseudo-labelled by taking the argmax of the outputs of the prediction network.

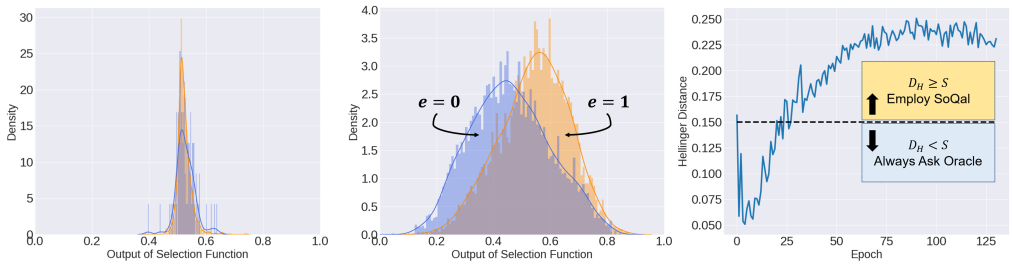
Proxy for misclassifications. Ideally, a network should only be reliant on an oracle when it misclassifies an instance itself. Quantifying these misclassifications is trivial in the presence of ground-truth labels. However, in the AL framework, we are interested in making decisions on *unlabelled* instances. Therefore, we need a reliable proxy for such misclassifications. We propose that this proxy be the output of the oracle selection network, t. For example, low and high values of t can indicate correct and incorrect network predictions, respectively. To learn this behaviour, h_θ needs an appropriate supervisory signal. For this, we choose the zero-one loss, e, of the prediction network, g_ϕ, as the ground-truth label (see Fig. 3). For a mini-batch of size, B, our objective function thus consists of two terms: 1) a cross-entropy class prediction loss for the main task, and 2) a binary cross-entropy oracle selection loss, with a weighting coefficient, β, (described next).

$$\mathcal{L} = \sum_{i=1}^{B} \overbrace{-\log\left(p(y_i = c|x_i, \omega, \phi)\right)}^{\text{class prediction loss}} \overbrace{-\beta e_i \log\left(h_\theta(t|x_i)\right) - (1 - e_i)\log\left(1 - h_\theta(t|x_i)\right)}^{\text{oracle selection loss}} \tag{4}$$

where c is the target class. During the early stages of training on *labelled* data, a network struggles to classify instances correctly. This means that the ratio of zero to one losses will be skewed toward the latter. As training progresses and the network becomes more adept at classifying instances, this ratio becomes skewed in the opposite direction. Such an imbalance in the ground-truth labels, e, and their subsequent shift send strong supervisory signals to h_θ. For example, during the early stages of training, the outputs of h_θ will be high and pulled towards $t = 1$ (high error) even if the corresponding instance was correctly classified. Such behaviour makes it difficult to ascertain whether instances have been misclassified, hindering the reliability of t as a proxy. To offset the aforementioned class imbalance, we introduce a dynamic hyperparameter, $\beta = \frac{\sum \delta_{e=0}}{\sum \delta_{e=1}}$, where δ is the Kronecker delta function. As training progress, $\beta < 1 \rightarrow \beta > 1$, as the ratio of correctly classified ($e = 0$) to misclassified ($e = 1$) instances within a mini-batch changes.

Decision-making with proxy. We aim to exploit t for the binary decision of either requesting a label from an oracle or generating a pseudo-label. One way to do so is via a simple threshold at 0.5. However, this threshold may not be appropriate. In designing a robust selection strategy, we must account for the distribution of the t values that corresponds to each decision and the separability of

such distributions. In Figs. 4a and 4b, we illustrate these distributions during the early and late stages of training, colour-coded based on whether the t values correspond to correctly-classified ($e = 0$) or misclassified ($e = 1$) *labelled* training instances.



(a) Early stage of training (b) Late stage of training (c) Hellinger distance

Figure 4: Density of the outputs, t, of g_θ colour-coded based on the zero-one classification error during the (a) early and (b) late stages of training. (c) the Hellinger distance, $\mathcal{D}_H$, between the two distributions of t during training increases as they become more separable.

We now outline how the binary decision is made. After each training epoch, we fit the t values in Fig. 4b to two unimodal Gaussian distributions. This generates $\mathcal{N}_0(\mu_0, \sigma_0^2)$ and $\mathcal{N}_1(\mu_1, \sigma_1^2)$ for $e = 0$ and $e = 1$, respectively. We choose to quantify the separability of these two distributions using the Hellinger distance, $\mathcal{D}_H \in [0, 1]$, as it allows for a straightforward threshold. Low separability expressed as $\mathcal{D}_H < S$ implies that h_θ has yet to generate a reliable proxy and thus an oracle is requested for a label. We note that the value of S can be altered depending on the relative level of trust one has in the network and oracle. When $\mathcal{D}_H \geq S$, we evaluate $\mathcal{N}_0$ and $\mathcal{N}_1$ at the t output for each acquired unlabelled instance and define $p(\mathrm{A})$ as the probability of requesting a label from an oracle. We elucidate the entire active learning algorithm in Appendix E.

$$p(\mathrm{A}) = \begin{cases} 1, & \mathcal{D}_H < S \\ 1, & \mathcal{N}\left(t|\mu_1, \sigma_1^2, e = 1\right) > \mathcal{N}\left(t|\mu_0, \sigma_0^2, e = 0\right) \text{ and } \mathcal{D}_H \geq S \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

5 Experimental Design

5.1 Datasets

We conduct experiments[1] in PyTorch (Paszke et al., 2019) on four publically-available datasets. These datasets consist of cardiac time-series data such as the photoplethysmogram (PPG) and the electrocardiogram (ECG) alongside cardiac arrhythmia labels. We use $\boldsymbol{\mathcal{D}_1}$ = PhysioNet 2015 PPG, $\boldsymbol{\mathcal{D}_2}$ = PhysioNet 2015 ECG (Clifford et al., 2015) (5-way), $\boldsymbol{\mathcal{D}_3}$ = PhysioNet 2017 ECG (Clifford et al., 2017) (4-way), $\boldsymbol{\mathcal{D}_4}$ = Cardiology ECG (Hannun et al., 2019) (12-way), and $\boldsymbol{\mathcal{D}_5}$ = CIFAR10 Krizhevsky et al. (2009) (10-way).

To observe the impact of the availability of labelled training data on the active learning procedure, we take a fraction $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$ of the training dataset and place it into the labelled set. Its complement is placed into the unlabelled set. Details about the data splits, preprocessing steps, and the network architecture can be found in Appendices F and G.

5.2 Baselines

Acquisition Functions. We compare our novel active learning framework to the state-of-the-art acquisition functions used in conjunction with MCD. These include **Var Ratio**, **Entropy**, and **BALD**, definitions of which can be found in Appendix A. We also compare to the scenario in which active learning is not employed (**No AL**).

[1]Our code is available at: https://github.com/danikiyasseh/SoCal

Selective Oracle Questioning. We experiment with baselines that exhibit varying degrees of oracle dependence. **No Oracle** is a scenario in which 0% of the labels that correspond to unlabelled instances are oracle-based and are instead pseudo-labelled by taking the argmax of the network predictions. **Epsilon Greedy** is a stochastic strategy Watkins (1989) that we adapt to exponentially decay the reliance of the network on an oracle as a function of the number of acquisition epochs. **Entropy Response** assumes that high entropy predictions generated by a network are indicative of instances that the network is unsure of. Therefore, we introduce a threshold, S_{Entropy}, such that if it is exceeded, an oracle is requested to label the chosen instance (see Appendix G). The most dependent baseline is **100% Oracle**, a traditionally-employed strategy in AL where 100% of the labels are oracle-based.

We do not compare our methods to Softmax Response (Geifman & El-Yaniv, 2017) and SelectiveNet (Geifman & El-Yaniv, 2019), despite their strong performance for selective classification, as they do not trivially extend to the setting in which labels are unavailable.

5.3 Hyperparameters

For all experiments, we chose the number of MC samples $T = 20$ to balance between computational complexity and accuracy of the approximation of the version space. We acquire unlabelled instances at pre-defined epochs during training which we refer to as acquisition epochs, $\tau = 5n$, $n \in \mathbb{N}^+$. During each acquisition epoch, we acquire $b = 2\%$ of the remaining unlabelled instances. We also investigate the effect of such hyperparameters on performance (see Appendices P-R). When experimenting with tracked acquisition functions, we chose the temporal period, $\Delta t = 1$. For the CIFAR10 experiments, we chose $T = 5$, $\tau = 2n$, and $b = 10\%$.

Selective Oracle Questioning. Recall that we delegate selective oracle questioning to the network only when $\mathcal{D}_H \geq S$. Given $\mathcal{D}_H$'s increasing trend during training (see Fig. 4c), we chose $S = 0.15$ to balance between the reliability of the proxy and the independence of the network from an oracle. We also explore the sensitivity of SoQal to this choice of S.

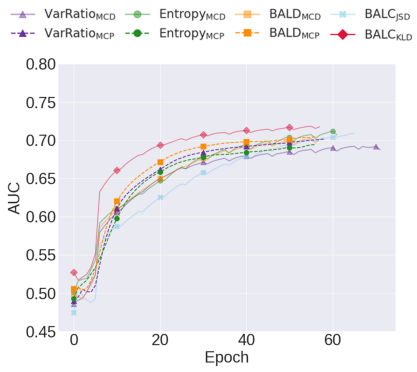
6 Experimental Results

6.1 Active Learning without Oracle

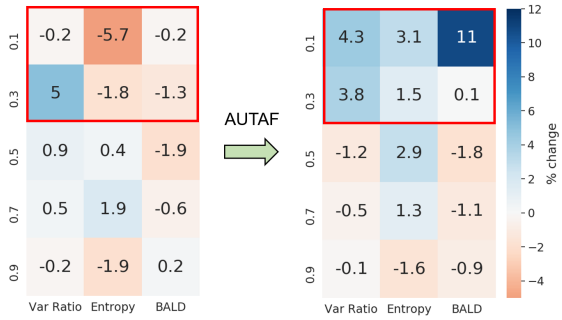
Active learning frameworks typically assume the presence of an oracle (expert annotator). However, oracles are not always available, particularly in the medical domain where physicians have limited time to provide annotations. To reflect this scenario, we evaluate the ability of our AL framework to operate *without* an oracle. In Fig. 5a, we illustrate the validation AUC of various methods when exposed to a fraction, β, of the labelled training data on $\mathcal{D}_2$ and $\mathcal{D}_5$, respectively.

We find that $\mathrm{BALC}_{\mathrm{KLD}}$ achieves strong generalization performance and does so in a fewer number of epochs relative to the remaining methods. For example, in Fig. 5a, $\mathrm{BALC}_{\mathrm{KLD}}$ achieves an AUC ≈ 0.69 after only 20 epochs whereas $\mathrm{BALD}_{\mathrm{MCD}}$ does so at epoch 40. This implies that $\mathrm{BALC}_{\mathrm{KLD}}$ can result in a two-fold reduction in training time. It also achieves a higher final AUC ≈ 0.72 relative to the remaining methods. We hypothesize that such behaviour is due to $\mathrm{BALC}_{\mathrm{KLD}}$'s improved ability to estimate the region of uncertainty, and thus acquires more informative instances. Moreover, given the absence of an oracle, these informative instances are likely to have also been pseudo-labelled correctly. The acquisition of more informative instances which are labelled correctly by the network suggests that such instances are *closer* to the decision boundary than their non-acquired counterparts yet are still on the correct side of the boundary. We arrive at similar conclusions for the remaining experiments (see Appendix H and J).

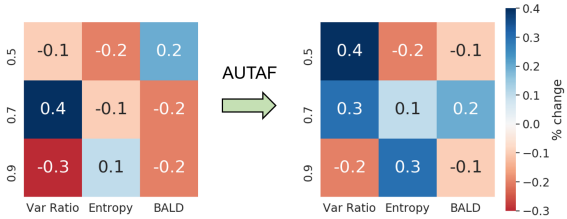
Having illustrated the potential benefit of static acquisition functions, we now move on to quantify the effect of incorporating temporal information on generalization performance. In Figs. 5b and 5c, we illustrate the percent change in the AUC when using MCP, with and without tracked acquisition functions, relative to MCD. We find that tracked acquisition functions are most useful when the initial size of the labelled dataset is small ($\downarrow \beta$ values) (red rectangle). For example, incorporating temporal information into BALD at $\beta = 0.1$ improves the generalization performance by 11%. We hypothesize that this improvement is due to the increased enumeration of hypotheses over time, which in turn, results in a more reliable approximation of the version space.



(a) $\mathcal{D}_2$ at $\beta = 0.3$



(b) Utility of tracked acquisition functions on $\mathcal{D}_1$



(c) Utility of tracked acquisition functions on $\mathcal{D}_5$

Figure 5: Validation AUC on **(a)** $\mathcal{D}_2$ at $\beta = 0.3$. Mean percent change in test AUC when comparing MCP with static and tracked acquisition functions to MCD with their static counterparts on **(b)** $\mathcal{D}_1$ and **(c)** $\mathcal{D}_5$. We show results for Var Ratio, Entropy, and BALD, at all fractions, $\beta \in [0.1, 0.3, 0.5, 0.7, 0.9]$ and across five seeds.

6.2 Active Learning with Noise-free Oracle

In this section, we relax the assumption that physicians are unavailable for annotation purposes. Instead, we focus on alleviating the labelling burden that is placed on physicians when they are available. More specifically, we assume that oracles can provide accurate labels, i.e., noise-free. In Table 1, we illustrate the test-set AUC of the oracle questioning strategies on all datasets at $\beta = 0.1$.

We find that SoQal consistently outperforms its counterparts across $\mathcal{D}_1$ - $\mathcal{D}_3$. For example, while using BALD_{MCD} on $\mathcal{D}_2$, SoQal achieves an $\text{AUC} = 0.707$ whereas Epsilon Greedy and Entropy

Table 1: Mean test AUC of oracle questioning strategies in the presence of a noise-free oracle. Results are shown for a subset of the acquisition functions on $\mathcal{D}_1 - \mathcal{D}_5$ and are averaged across five seeds.

Dataset	Ac. Function α	Oracle Questioning Method					No AL
		No Oracle	Entropy Response	Epsilon Greedy	SoQal (ours)	100% Oracle	
$\mathcal{D}_1$	BALD_{MCD}	0.465 ± 0.017	0.496 ± 0.039	0.491 ± 0.028	$\mathbf{0.621 \pm 0.021}$	0.653 ± 0.013	0.577 ± 0.014
	BALD_{MCP}	0.464 ± 0.023	0.517 ± 0.043	0.501 ± 0.043	$\mathbf{0.645 \pm 0.015}$	0.676 ± 0.020	
	BALC_{KLD}	0.500 ± 0.023	0.548 ± 0.034	0.548 ± 0.042	$\mathbf{0.598 \pm 0.055}$	0.634 ± 0.030	
	Temporal BALC_{KLD}	0.496 ± 0.024	0.536 ± 0.040	0.521 ± 0.059	$\mathbf{0.646 \pm 0.067}$	0.659 ± 0.033	
$\mathcal{D}_2$	BALD_{MCD}	0.573 ± 0.063	0.584 ± 0.041	0.609 ± 0.071	$\mathbf{0.707 \pm 0.038}$	0.713 ± 0.053	0.679 ± 0.040
	BALD_{MCP}	0.589 ± 0.045	0.638 ± 0.043	0.637 ± 0.044	$\mathbf{0.677 \pm 0.042}$	0.735 ± 0.028	
	BALC_{KLD}	0.602 ± 0.044	0.582 ± 0.017	0.643 ± 0.033	$\mathbf{0.677 \pm 0.024}$	0.722 ± 0.018	
	Temporal BALC_{KLD}	0.575 ± 0.017	0.612 ± 0.050	0.605 ± 0.019	$\mathbf{0.648 \pm 0.057}$	0.735 ± 0.011	
$\mathcal{D}_3$	BALD_{MCD}	0.581 ± 0.014	0.588 ± 0.013	0.673 ± 0.015	$\mathbf{0.721 \pm 0.025}$	0.802 ± 0.008	0.716 ± 0.012
	BALD_{MCP}	0.623 ± 0.020	0.676 ± 0.058	0.665 ± 0.028	$\mathbf{0.720 \pm 0.044}$	0.798 ± 0.007	
	BALC_{KLD}	0.631 ± 0.010	0.629 ± 0.004	0.643 ± 0.041	$\mathbf{0.731 \pm 0.033}$	0.787 ± 0.008	
	Temporal BALC_{KLD}	0.600 ± 0.005	0.630 ± 0.014	0.654 ± 0.019	$\mathbf{0.730 \pm 0.024}$	0.794 ± 0.002	
$\mathcal{D}_4$	BALD_{MCD}	0.486 ± 0.011	0.489 ± 0.030	0.474 ± 0.037	0.468 ± 0.021	0.585 ± 0.011	0.486 ± 0.023
	BALD_{MCP}	0.493 ± 0.030	0.504 ± 0.026	0.492 ± 0.024	0.499 ± 0.029	0.605 ± 0.024	
	BALC_{KLD}	0.505 ± 0.032	0.504 ± 0.039	0.473 ± 0.010	0.495 ± 0.012	0.588 ± 0.033	
	Temporal BALC_{KLD}	0.511 ± 0.030	0.496 ± 0.023	0.496 ± 0.023	0.503 ± 0.010	0.532 ± 0.027	
$\mathcal{D}_5$	BALD_{MCD}	0.891 ± 0.003	0.893 ± 0.002	0.901 ± 0.001	0.899 ± 0.003	0.902 ± 0.003	0.898 ± 0.002
	BALD_{MCP}	0.895 ± 0.002	0.894 ± 0.003	0.904 ± 0.003	0.897 ± 0.003	0.900 ± 0.002	
	BALC_{KLD}	0.894 ± 0.003	0.895 ± 0.001	0.899 ± 0.002	0.895 ± 0.001	0.899 ± 0.002	
	Temporal BALC_{KLD}	0.895 ± 0.001	0.894 ± 0.002	0.899 ± 0.002	0.896 ± 0.002	0.899 ± 0.004	

Response achieve AUC $= 0.609$ and 0.584, respectively. Such a finding suggests that SoQal is better equipped to know *when* and for which *instance* a label should be requested from an oracle. One could argue that SoQal's superiority is due to its high dependence on the oracle. In fact, we show that this is not the case in Appendix L. On the other hand, we observe that SoQal performs on par with the other methods on $\mathcal{D}_4$ and $\mathcal{D}_5$. We hypothesize that this outcome is due to the cold-start problem (Konyushkova et al., 2017) where AL algorithms fail to learn due to the limited availability of labelled training data. We support this claim with experiments in Appendix M. Moreover, we remind readers that by increasing the value of S in the SoQal experiments, networks can cede more control to the oracle and thus further improve performance, an effect we quantify in Appendix N.

6.3 Active Learning with Noisy Oracle

So far, we have presented scenarios in which oracles are either absent or present with the ability to provide accurate labels. In healthcare, however, physicians may be ill-trained, fatigued, or unable to diagnose a case due to its difficulty. We simulate this scenario by introducing two types of label noise. We stochastically flip each label to 1) any other label randomly (**Random**), or 2) its nearest neighbour from a *different* class in a compressed subspace (**Nearest Neighbour**). Whereas the first form of noise is extreme, the latter is more realistic as it may represent uncertain physician diagnoses. To simulate various magnitudes of noise, we chose the probability of introducing noise, $\gamma = [0.05, 0.1, 0.2, 0.4, 0.8]$. In Fig. 6, we illustrate the effect of label noise on the test AUC for the various oracle questioning strategies.

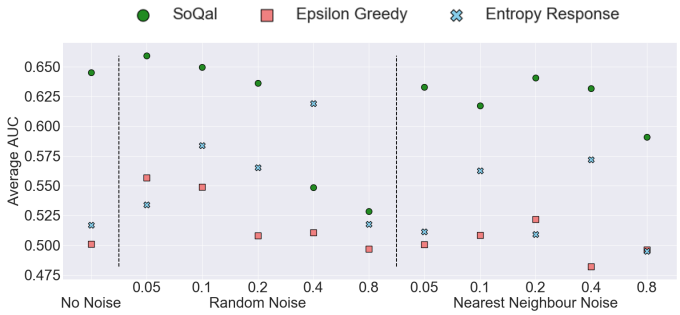


Figure 6: Average test AUC for the oracle questioning strategies in the absence and presence, of various magnitudes, of label noise on $\mathcal{D}_1$ using BALD_{MCP}. We show that with up to 80% random or nearest neighbour label noise, SoQal still outperforms the remaining methods when trained *without* label noise. This illustrates that SoQal is better equipped to deal with oracle label noise.

We find that SoQal outperforms the remaining strategies regardless of noise type and magnitude (except at 40% random noise). For example, at 5% random noise, SoQal achieves an AUC ≈ 0.66 whereas Epsilon Greedy and Entropy Response achieve an AUC ≈ 0.56 and ≈ 0.53, respectively. Surprisingly, we find that the introduction of label noise can sometimes improve performance. For example, SoQal's AUC $\approx 0.64 \rightarrow 0.66$ with no noise and 5% random noise, respectively. We hypothesize that this is due to inherent label noise in the datasets. By introducing further noise, we nudge these labels towards their ground-truth values. Moreover, SoQal is better able to deal with label noise than its counterparts. Specifically, SoQal at 80% random noise achieves AUC ≈ 0.53 whereas Epsilon Greedy and Entropy Response trained *without noise* achieve AUC ≈ 0.50 and ≈ 0.52, respectively. This effect, which is even more pronounced when dealing with nearest neighbour noise, indicates the utility of SoQal in the presence of a noisy oracle. We arrive at similar conclusions when experimenting with other datasets and acquisition functions (see Appendix O).

7 Discussion and Future Work

In this paper, we proposed a novel consistency-based active learning framework which perturbs both inputs and network parameters and acquires instances to which the network is least robust. We illustrate the utility of this approach in the *absence* of an oracle. Moreover, we propose a strategy that dynamically determines whether an oracle should be requested for a label. We empirically show that

this approach is better able to deal with noisy oracles than the baseline methods. We now elucidate several future avenues worth exploring.

Incorporating prior information. The default mode for SoQal is deferral to an oracle. However, relevant a priori information, such as the degree of noise inherent in the oracle's labels, can be incorporated to alter either the default mode or the Hellinger threshold, S.

Incorporating multiple oracles. In this work, we focused on the presence of a single oracle. Scenarios in which multiple oracles exist may better reflect clinical environments which include multiple experts of various levels of competency. Therefore, dynamically querying these oracles might be of interest.

REFERENCES

Zachi I Attia, Suraj Kapa, Francisco Lopez-Jimenez, Paul M McKie, Dorothy J Ladewig, Gaurav Satam, Patricia A Pellikka, Maurice Enriquez-Sarano, Peter A Noseworthy, Thomas M Munger, et al. Screening for cardiac contractile dysfunction using an artificial intelligence–enabled electrocardiogram. *Nature Medicine*, 25(1):70–74, 2019.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

C Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1):41–46, 1970.

Gari D Clifford, Ikaro Silva, Benjamin Moody, Qiao Li, Danesh Kella, Abdullah Shahin, Tristan Kooistra, Diane Perry, and Roger G Mark. The physionet/computing in cardiology challenge 2015: reducing false arrhythmia alarms in the icu. In *2015 Computing in Cardiology Conference*, pp. 273–276, 2015.

Gari D Clifford, Chengyu Liu, Benjamin Moody, H Lehman Li-wei, Ikaro Silva, Qiao Li, AE Johnson, and Roger G Mark. Af classification from a short single lead ECG recording: the physionet/computing in cardiology challenge 2017. In *2017 Computing in Cardiology*, pp. 1–4, 2017.

David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

Corinna Cortes, Giulia DeSalvo, and Mehryar Mohri. Learning with rejection. In *International Conference on Algorithmic Learning Theory*, pp. 67–82. Springer, 2016.

Ofer Dekel, Claudio Gentile, and Karthik Sridharan. Selective sampling and active learning from single and multiple teachers. *Journal of Machine Learning Research*, 13(Sep):2655–2697, 2012.

Ran El-Yaniv and Yair Wiener. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11(May):1605–1641, 2010.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international Conference on Machine Learning*, pp. 1050–1059, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep Bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1183–1192. JMLR. org, 2017.

Mingfei Gao, Zizhao Zhang, Guo Yu, Sercan O Arik, Larry S Davis, and Tomas Pfister. Consistency-based semi-supervised active learning: towards minimizing labeling cost. *arXiv preprint arXiv:1910.07153*, 2019.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in neural information processing systems*, pp. 4878–4887, 2017.

Yonatan Geifman and Ran El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. *arXiv preprint arXiv:1901.09192*, 2019.

Wenbo Gong, Sebastian Tschiatschek, Richard Turner, Sebastian Nowozin, and José Miguel Hernández-Lobato. Icebreaker: element-wise active information acquisition with bayesian deep latent gaussian model. *arXiv preprint arXiv:1908.04537*, 2019.

Awni Y Hannun, Pranav Rajpurkar, Masoumeh Haghpanahi, Geoffrey H Tison, Codie Bourn, Mintu P Turakhia, and Andrew Y Ng. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25(1):65, 2019.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Learning active learning from data. In *Advances in Neural Information Processing Systems*, pp. 4225–4235, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Ziyin Liu, Zhikang Wang, Paul Pu Liang, Russ R Salakhutdinov, Louis-Philippe Morency, and Masahito Ueda. Deep gamblers: Learning to abstain with portfolio theory. In *Advances in Neural Information Processing Systems*, pp. 10622–10632, 2019.

Andrew Kachites McCallumzy and Kamal Nigamy. Employing em and pool-based active learning for text classification. In *Proc. International Conference on Machine Learning*, pp. 359–367, 1998.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

Ryan Poplin, Avinash V Varadarajan, Katy Blumer, Yun Liu, Michael V McConnell, Greg S Corrado, Lily Peng, and Dale R Webster. Prediction of cardiovascular risk factors from retinal fundus photographs via deep learning. *Nature Biomedical Engineering*, 2(3):158, 2018.

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

Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5972–5981, 2019.

Asim Smailagic, Pedro Costa, Hae Young Noh, Devesh Walawalkar, Kartik Khandelwal, Adrian Galdran, Mostafa Mirshekari, Jonathon Fagert, Susu Xu, Pei Zhang, et al. Medal: Accurate and robust deep active learning for medical image analysis. In *IEEE International Conference on Machine Learning and Applications*, pp. 481–488, 2018.

Asim Smailagic, Pedro Costa, Alex Gaudio, Kartik Khandelwal, Mostafa Mirshekari, Jonathon Fagert, Devesh Walawalkar, Susu Xu, Adrian Galdran, Pei Zhang, et al. O-medal: Online active deep learning for medical image analysis. *arXiv preprint arXiv:1908.10508*, 2019.

Nenad Tomašev, Xavier Glorot, Jack W Rae, Michal Zielinski, Harry Askham, Andre Saraiva, Anne Mottram, Clemens Meyer, Suman Ravuri, Ivan Protsyuk, et al. A clinically applicable approach to continuous prediction of future acute kidney injury. *Nature*, 572(7767):116–119, 2019.

Ruth Urner, Shai Ben David, and Ohad Shamir. Learning from weak teachers. In *Artificial intelligence and statistics*, pp. 1252–1260, 2012.

Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. *arXiv preprint arXiv:1903.03825*, 2019.

Guijin Wang, Chenshuang Zhang, Yongpan Liu, Huazhong Yang, Dapeng Fu, Haiqing Wang, and Ping Zhang. A global and updatable ecg beat classification system based on recurrent neural networks and active learning. *Information Sciences*, 501:523–542, 2019.

Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

Yair Wiener and Ran El-Yaniv. Agnostic selective classification. In *Advances in Neural Information Processing Systems*, pp. 1665–1673, 2011.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. 2019.

Songbai Yan, Kamalika Chaudhuri, and Tara Javidi. Active learning from imperfect labelers. In *Advances in Neural Information Processing Systems*, pp. 2128–2136, 2016.

Chicheng Zhang and Kamalika Chaudhuri. Active learning from weak and strong labelers. In *Advances in Neural Information Processing Systems*, pp. 703–711, 2015.

Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison, Department of Computer Sciences, 2005.

A ACQUISITION FUNCTIONS α

$$\begin{aligned} \text{Variance Ratio} &= 1 - \frac{1}{T}\sum_{t=1}^{T}\left(\delta\left(\underset{c}{\text{argmax}}\, p(y=c|x,\omega_t) = \hat{c}\right)\right) \\ \hat{c} &= \underset{c}{\text{argmax}}\left(\underset{c}{\text{argmax}}\, p(y=c|x,\omega_t)\ \forall t \in (1,T)\right) \end{aligned} \tag{6}$$

where $\hat{c}$ is the most common class prediction across the T MC samples and δ is the Dirac delta function that evaluates to 1 if its argument is true, and 0 otherwise.

$$\text{Entropy H} = -\sum_{c=1}^{C} p(y=c|x)\log p(y=c|x) \tag{7}$$

$$\begin{aligned} \text{BALD} &= \text{JSD}(p_1, p_2, \ldots, p_T) \\ &= \text{H}(p(y|x)) - \mathbb{E}_{p(w|D_{train})}\left[\text{H}(p(y|x,w))\right] \end{aligned} \tag{8}$$

where C is the number of classes in the task formulation and $p(y=c|x,\omega)$ is the probability assigned by a network parameterised by ω to a particular class c when given an input x.

B Derivation of Monte Carlo Perturbations

$$\begin{aligned} \text{BALD}_{\text{MCP}} &= \text{JSD}(p_1, p_2, \ldots, p_T) \\ &= \text{H}(p(y|x)) - \mathbb{E}_{p(z|D_{train})}\left[\text{H}(p(y|x,z))\right] \end{aligned} \tag{9}$$

$$\begin{aligned} \text{H}(p(y|x)) &= \text{H}\left(\int p(y|z)p(z|x)dz\right) \\ &= \text{H}\left(\int p(y|z)q_\phi(z|x)dz\right) \\ &\approx \text{H}\left(\frac{1}{T}\sum_{t=1}^{T} p(y|\hat{z}_t)\right) \end{aligned} \tag{10}$$

where z represents the perturbed input, T is the number of Monte Carlo samples, and $\hat{z}_t \sim q_\phi(z|x)$ is a sample from some perturbation generator.

$$\begin{aligned} \mathbb{E}_{p(z|D_{train})}\left[\text{H}(p(y|x,z))\right] &= \mathbb{E}_{q_\phi(z|x)}\left[\text{H}(p(y|x,z))\right] \\ &\approx \frac{1}{T}\sum_{t=1}^{T}\left[\text{H}(p(y|\hat{z}_t))\right] \\ &= \frac{1}{T}\sum_{t=1}^{T}\left[-\sum_{c=1}^{C} p(y=c|\hat{z}_t)\log p(y=c|\hat{z}_t)\right] \end{aligned} \tag{11}$$

C Derivation of Bayesian Active Learning by Consistency

$$\text{BALC}_{\text{JSD}} = \mathbb{E}_{p(\omega|D_{train})}\left[\mathcal{D}_{KL}(p(y|x,\omega) \parallel p(y|z,\omega))\right] - \mathcal{D}_{KL}(p(y|x) \parallel p(y|z)) \tag{12}$$

where z is the perturbed version of the input and $\mathcal{D}_{KL}$ is the Kullback-Leibler divergence.

$$\begin{aligned} \mathbb{E}_{p(\omega|D_{train})}\left[\mathcal{D}_{KL}(p(y|x,\omega) \parallel p(y|z,\omega))\right] &= \mathbb{E}_{q_\theta(\omega)}\left[\mathcal{D}_{KL}(p(y|x,\omega) \parallel p(y|z,\omega))\right] \\ &\approx \frac{1}{T}\sum_{t=1}^{T}\left[\mathcal{D}_{KL}(p(y|x,\hat{\omega}_t \parallel p(y|\hat{x},\hat{\omega}_t))\right] \\ &= \frac{1}{T}\sum_{t=1}^{T}\left[\sum_{c=1}^{C} p(y=c|x,\hat{\omega}_t)\log\frac{p(y=c|x,\hat{\omega}_t)}{p(y=c|z,\hat{\omega}_t)}\right] \end{aligned} \tag{13}$$

$$\begin{aligned} \mathcal{D}_{KL}(p(y|x)||p(y|z)) &= \mathcal{D}_{KL}\left(\int p(y|\omega,x)p(\omega)d\omega \,\middle\|\, \int p(y|\omega,z)p(\omega)d\omega\right) \\ &= \mathcal{D}_{KL}\left(\int p(y|\omega,x)q_\theta(\omega)d\omega \,\middle\|\, \int p(y|\omega,z)q_\theta(\omega)d\omega\right) \\ &\approx \mathcal{D}_{KL}\left(\frac{1}{T}\sum_{t=1}^{T} p(y|\hat{\omega}_t,x) \,\middle\|\, \frac{1}{T}\sum_{t=1}^{T} p(y|\hat{\omega}_t,z)\right) \\ &= \frac{1}{C}\sum_{c=1}^{C}\left[\frac{1}{T}\sum_{t=1}^{T} p(y=c|\hat{\omega}_t,x)\log\frac{\frac{1}{T}\sum_{t=1}^{T} p(y=c|\hat{\omega}_t,x)}{\frac{1}{T}\sum_{t=1}^{T} p(y=c|\hat{\omega}_t,\hat{x})}\right] \end{aligned} \tag{14}$$

where the integral is approximated by T Monte Carlo samples, $\hat{\omega} \sim q_\theta(w)$ represents the parameters sampled from the Monte Carlo distribution, and C represents the number of classes in the task formulation.

D CHERNOFF BOUND ON ERROR RATE OF SELECTION NETWORK

D.1 DERIVATION OF CHERNOFF BOUND

In this section, we derive the Chernoff bound as an upper bound on the binary classification error, $P(\text{error})$, of the oracle selection network h_θ. This helps determine how reliable the output of h_θ is as a proxy for the classification performance of the main task.

$$
\begin{aligned}
P(\text{error}) &= \int_{-\infty}^{\infty} P(\text{error}|x)p(x)dx \\
&= \int_{-\infty}^{\infty} \min\left[P(y=0|x), P(y=1|x)\right] p(x)dx \\
&\leq \int_{-\infty}^{\infty} P(y=0|x)^{\beta} P(y=1|x)^{1-\beta} p(x)dx \\
&= P(y=0)^{\beta} P(y=1)^{1-\beta} \int_{-\infty}^{\infty} P(x|y=0)^{\beta} P(x|y=1)^{1-\beta} dx \\
&= P(y=0)^{\beta^*} P(y=1)^{1-\beta^*} e^{-\left[\frac{\beta^*(1-\beta^*)(\mu_0-\mu_1)^2}{2(\beta^*\sigma_0^2+(1-\beta^*)\sigma_1^2)} + \frac{1}{2} log \frac{\beta^*\sigma_0^2+(1-\beta^*)\sigma_1^2}{\sigma_0^{2\beta^*}\sigma_1^{2(1-\beta^*)}}\right]}
\end{aligned}
\tag{15}
$$

In order to calculate β^*, we minimize the following term using the Broyden-Fletcher-Goldfarb-Shannon (BFGS) algorithm with an initial value of $\beta_0 = 0$.

$$
\beta^* = \text{argmin}_\beta - \left[\frac{\beta(1-\beta)(\mu_0-\mu_1)^2}{2(\beta\sigma_0^2+(1-\beta)\sigma_1^2)} + \frac{1}{2} ln \frac{\beta\sigma_0^2+(1-\beta)\sigma_1^2}{\sigma_0^{2\beta}\sigma_1^{2(1-\beta)}}\right] \tag{16}
$$

E ALGORITHMS

E.1 BAYESIAN ACTIVE LEARNING BY CONSISTENCY

Algorithm 1 illustrates the BALC procedure with the option of incorporating temporal information shown in blue.

Algorithm 1: Bayesian Active Learning by Consistency

Input: acquisition epochs τ, temporal period Δt, labelled data $\mathcal{L}$, unlabelled data $\mathcal{U}$, network parameters ω, MC samples T, acquisition percentage b
while training **do**
  if epoch in Δt **then**
    for $x \sim \mathcal{U}$ **do**
      $z = x + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$
      for MC sample in T **do**
        obtain $p(y|x, \omega)$
        obtain $p(y|z, \omega)$
      end for
      calculate α using eq. 2 or eq. 1
      $\alpha(t) = \alpha$
    end for
  end if
  if epoch in τ **then**
    calculate α using eq. 3
    SortDescending(α)
    $\mathcal{U}_b \subseteq \mathcal{U}$
    $\mathcal{U} \in (\mathcal{U} \setminus \mathcal{U}_b)$
    $\mathcal{L} \in (\mathcal{L} \cup \mathcal{U}_b)$
  end if
end while

E.2 SOQAL

In this section, we outline the algorithm for performing SoQal. More specifically, Algorithm 2 presents the generic framework for the active learning procedure. Algorithm 3 elucidates the exact steps required to perform selective oracle questioning using SoQal.

Algorithm 2: Active Learning Procedure

Input: acquisition epochs τ, temporal period Δt, labelled data $\mathcal{L}$, unlabelled data $\mathcal{U}$, network parameters ω, MC samples T, acquisition percentage b
while training **do**
  if epoch in Δt **then**
    for $x \sim \mathcal{U}$ **do**
      for MC sample in T **do**
        obtain $p(y|x, \omega)$
      end for
      calculate α
    end for
  end if
  if epoch in τ **then**
    SortDescending(α)
    $x_b \subset X_U$
    $y_b = \text{SoQal}(x_b)$
    $\mathcal{U} \in (\mathcal{U} \setminus (x_b, y_b))$
    $\mathcal{L} \in (\mathcal{L} \cup (x_b, y_b))$
  end if
end while

Algorithm 3: SoQal

Input: unlabelled inputs x_b, Hellinger distance $\mathcal{D}_H$, Hellinger threshold S
for $x \sim x_b$ **do**
  $o = g_\theta(x)$
  if $\mathcal{D}_H > S$ **then**
    calculate p(asking oracle) from eq. 5
    if p(asking oracle) $= 1$ **then**
      $y_b \subset Y_U$
    else
      $y_b = \text{argmax} p(y|x, \omega)$
    end if
  end if
end for

F DATASETS

F.1 DATA PREPROCESSING

Each dataset consists of cardiac time-series waveforms alongside their corresponding cardiac arrhythmia label. Each waveform was split into non-overlapping frames of 2500 samples.

PhysioNet 2015 PPG, $\mathcal{D}_1$ (Clifford et al., 2015). This dataset consists of photoplethysmogram (PPG) time-series waveforms sampled at 250Hz and five cardiac arrhythmia labels: Asystole, Extreme Bradycardia, Extreme Tachycardia, Ventricular Tachycardia, and Ventricular Fibrillation. Only patients with a True Positive Alarm are considered. The PPG frames were normalized in amplitude between the values of 0 and 1.

PhysioNet 2015 ECG, $\mathcal{D}_2$ (Clifford et al., 2015). This dataset consists of electrocardiogram (ECG) time-series waveforms sampled at 250Hz and five cardiac arrhythmia labels: Asystole, Extreme Bradycardia, Extreme Tachycardia, Ventricular Tachycardia, and Ventricular Fibrillation. Only patients with a True Positive Alarm are considered. The ECG frames were normalized in amplitude between the values of 0 and 1.

PhysioNet 2017 ECG, $\mathcal{D}_3$ (Clifford et al., 2017). This dataset consists of ECG time-series waveforms sampled at 300Hz and four labels: Normal, Atrial Fibrillation, Other, and Noisy. The ECG frames were not normalized.

Cardiology ECG, $\mathcal{D}_4$ (Hannun et al., 2019). This dataset consists of ECG time-series waveforms sampled at 200Hz and twelve cardiac arrhythmia labels: Atrial Fibrillation, Atrio-ventricular Block, Bigeminy, Ectopic Atrial Rhythm, Idioventricular Rhythm, Junctional Rhythm, Noise, Sinus Rhythm, Supraventricular Tachycardia, Trigeminy, Ventricular Tachycardia, and Wenckebach. Sudden bradycardia cases were excluded from the data as they were not included in the original formulation by the authors. The ECG frames were not normalized.

CIFAR10, $\mathcal{D}_5$ (Krizhevsky et al., 2009). This dataset consists of 60,000 colour images of dimension 32×32 associated with 10 classes. These classes are $\mathrm{airplane}$, $\mathrm{automobile}$, bird, cat, deer, dog, frog, horse, ship, and truck. Each image was normalized between the range of -1 and 1. No data augmentation was applied during the training procedure.

F.2 Data Samples

All datasets were split into training, validation, and test sets according to patient ID using a 60, 20, 20 configuration. In other words, patients appeared in only one of the sets. Samples in the training set were further split into a labelled and an unlabelled subset, also according to patient ID. In Tables 2 and 3, we show the number of samples and patients used in each of these sets.

F.2.1 Consistency-based Active Learning Experiments

Table 2: Sample sizes (number of patients for cardiac datasets) of train/val/test splits. Datasets $\mathcal{D}_1$ to $\mathcal{D}_4$ are defined in Sec. 5.1 of the main manuscript.

Dataset	Fraction β	Train Labelled	Train Unlabelled	Val	Test
$\mathcal{D}_1$	0.1	401 (18)	4,233 (171)	1,124 (47)	1,435 (58)
	0.3	1,285 (55)	3,349 (134)		
	0.5	2,187 (92)	2,447 (97)		
	0.7	3,132 (129)	1,502 (60)		
	0.9	4,184 (166)	450 (23)		
$\mathcal{D}_2$	0.1	401 (18)	4,233 (171)	1,124 (47)	1,435 (58)
	0.3	1,285 (55)	3,349 (134)		
	0.5	2,187 (92)	2,447 (97)		
	0.7	3,132 (129)	1,502 (60)		
	0.9	4,184 (166)	450 (23)		
$\mathcal{D}_3$	0.1	1,776 (545)	16,479 (4,914)	4,582 (1,364)	5,824 (1705)
	0.3	5,399 (1636)	12,856 (3,823)		
	0.5	9,054 (2727)	9,201 (2,732)		
	0.7	12,733 (3818)	5,522 (1,641)		
	0.9	16,365 (4909)	1,890 (550)		
$\mathcal{D}_4$	0.1	452 (20)	4,110 (181)	1,131 (50)	1,386 (62)
	0.3	1,368 (60)	3,194 (141)		
	0.5	2,280 (101)	2,282 (100)		
	0.7	3,200 (140)	1,362 (61)		
	0.9	4,079 (180)	483 (21)		
$\mathcal{D}_5$	0.5	20,000	20,000	10,000	10,000
	0.7	28,000	12,000		
	0.9	36,000	4,000		

F.2.2 Selective Oracle Questioning Experiments

Table 3: Sample sizes (number of patients) of training, validation, and test sets.

Dataset	Training Labelled	Training Unlabelled	Validation	Test
$\mathcal{D}_1$	401 (18)	4233 (171)	1124 (47)	1435 (58)
$\mathcal{D}_2$	401 (18)	4233 (171)	1124 (47)	1435 (58)
$\mathcal{D}_3$	1,776 (545)	16,479 (4,914)	4,582 (1,364)	5,824 (1,705)
$\mathcal{D}_4$	452 (20)	4,110 (181)	1,131 (50)	1,386 (62)
$\mathcal{D}_5$	676 (12)	5,880 (116)	1,566 (32)	1,971 (40)

G Implementation Details

In this section, we outline the network architecture used for all experiments conducted in the main manuscript. We also outline the batchsize and learning rate associated with training on each of the datasets.

G.1 Network Architecture

Table 4: Network architectures used for time-series and image experiments. K, C_{in}, and C_{out} represent the kernel size, number of input channels, and number of output channels, respectively. A stride of 3 and 1 was used for Conv1D and Conv2D operators, respectively.

(a) Network for time-series datasets

Layer Number	Layer Components	Kernel Dimension
1	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	7 x 1 x 4 (K x C_{in} x C_{out})
2	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	7 x 4 x 16
3	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	7 x 16 x 32
4	Linear ReLU	320 x 100
5	Linear	100 x C (classes)

(b) Network for CIFAR10

Layer Number	Layer Components	Kernel Dimension
1	Conv 2D ReLU MaxPool(2)	5 x 3 x 6
2	Conv 2D ReLU MaxPool(2)	5 x 6 x 16
3	Linear ReLU Dropout(0.1)	160 x 120
4	Linear ReLU Dropout(0.1)	120 x 84
5	Linear	84 x C (classes)

G.2 EXPERIMENT DETAILS

Table 5: Batchsize and learning rates used for training with different datasets. The Adam optimizer was used for all experiments.

Dataset	Batchsize	Learning Rate
$\mathcal{D}_1$	256	10^{-4}
$\mathcal{D}_2$	256	10^{-4}
$\mathcal{D}_3$	256	10^{-4}
$\mathcal{D}_4$	16	10^{-4}

G.3 PERTURBATION DETAILS

When conducting the MCP and BALC experiments, we perturbed each of the time-series frames with additive Gaussian noise, $\epsilon \sim \mathcal{N}(0, \sigma)$ where we chose σ based on the specific dataset to avoid introducing too much noise. For CIFAR10, we perturbed the images by applying a sequence of data augmentation steps inspired by work in Chen et al. (2020). The details of these perturbations can be found in Table 6. We applied all perturbations to the input data before normalization.

Table 6: Perturbations applied to different datasets during MCP and BALC implementations. p represents the probability of applying a particular augmentation method.

Dataset	Perturbation
$\mathcal{D}_1$	$\epsilon \sim \mathcal{N}(0, 100)$
$\mathcal{D}_2$	$\epsilon \sim \mathcal{N}(0, 100)$
$\mathcal{D}_3$	$\epsilon \sim \mathcal{N}(0, 100)$
$\mathcal{D}_4$	$\epsilon \sim \mathcal{N}(0, 100)$
$\mathcal{D}_5$	1) $\mathrm{RandomResizedCrop}(\mathrm{scale} = (0.8, 1.0))$ 2) $\mathrm{RandomApply}(\mathrm{ColorJitter}(0.8,0.8,0.8,0.2), p = 0.2)$ 3) $\mathrm{RandomGrayscale}(p = 0.2)$

G.4 BASELINE IMPLEMENTATIONS

In this section, we outline our implementation of the baseline methods used in the selective oracle questioning experiments.

G.4.1 ENTROPY RESPONSE

This approach is anchored around the idea that network outputs that exhibit high entropy (i.e., close to a uniform distribution) are likely to correspond to instances that the network is uncertain of. Consequently, we exploited this idea to determine whether a label is requested from an oracle or if a pseudo-label should be generated instead. More specifically, we introduced a threshold, $\mathrm{S}_{Entropy} = w \times \mathrm{S}_{Max}$, which is a fraction of the maximum entropy possible for a particular classification problem. As mentioned, $\mathrm{S}_{Max} = \log C$, where C is the number of classes. We chose $w = 0.9$ to balance between oracle dependence and pseudo-label accuracy. This value was kept fixed during training. In our implementation, we take the mean of the network outputs as a result of the perturbations, calculate its entropy, and determine whether it exceeds the aforementioned threshold. If it does, then the uncertainty is deemed high and a label is requested from an oracle.

G.4.2 EPSILON GREEDY

This approach is inspired by the reinforcement learning literature and is used to decay the dependence of network on the oracle. More specifically, we define $\epsilon = e^{\frac{-\text{epoch}}{k \times \tau}}$ where epoch represents the training epoch number and τ is the epoch interval at which acquisitions are performed. ϵ decays from $1 \rightarrow 0$ as training progresses. We chose $k = \tau = 5$ in order to balance between oracle dependence and pseudo-label accuracy. To determine whether a label is requested from an oracle, we generate a random number, $R \sim \mathcal{U}(0, 1)$, for a uniform distribution and check whether it is below ϵ. If this is satisfied, then an oracle is requested for a label, and a pseudo-label is generated otherwise. As designed, this approach starts off with 100% dependence on an oracle and decays towards minimal dependence as training progresses.

H TEST SET PERFORMANCE IN THE ABSENCE OF ORACLE

In this section, we quantify and compare the performance of our consistency-based active learning framework to state-of-the-art AL methods on four diverse datasets, $\mathcal{D}_1$ - $\mathcal{D}_4$ and for a range of fraction values, $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$. Across Tables 7 - 11, we show that our method outperforms the baseline methods in 17 out of 28 (61%) experimental categories. Moreover, in half of all experimental categories, temporal acquisition functions perform best.

H.1 PHYSIONET 2015 PPG, $\mathcal{D}_1$

Table 7: Mean test Set AUC on $\mathcal{D}_1$. Bolded elements represent the best performing method and acquisition function α for each fraction β. No AL represents training without an active learning strategy. Results shown across 5 seeds.

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.1	MCD	0.476 ± 0.022	0.475 ± 0.020	0.465 ± 0.017	-	0.468 ± 0.032	0.492 ± 0.022	0.476 ± 0.015	-	0.577 ± 0.014
	MCP	0.475 ± 0.037	0.448 ± 0.019	0.464 ± 0.023	-	0.497 ± 0.028	0.490 ± 0.032	**0.515 ± 0.025**	-	
	$BALC_{JSD}$	-	-	-	0.511± 0.031	-	-	-	0.494 ± 0.021	
	$BALC_{KLD}$	-	-	-	0.500 ± 0.023	-	-	-	0.496 ± 0.024	
0.3	MCD	0.603 ± 0.021	0.618 ± 0.026	0.606 ± 0.032	-	0.607 ± 0.012	0.614 ± 0.009	0.617± 0.035	-	0.653 ± 0.017
	MCP	0.633 ± 0.024	0.607 ± 0.015	0.598 ± 0.015	-	0.626 ± 0.032	0.627 ± 0.026	0.606 ± 0.031	-	
	$BALC_{JSD}$	-	-	-	0.594 ± 0.009	-	-	-	0.600 ± 0.016	
	$BALC_{KLD}$	-	-	-	**0.633± 0.017**	-	-	-	0.617 ± 0.019	
0.5	MCD	0.650 ± 0.011	0.650 ± 0.011	0.660 ± 0.013	-	0.654 ± 0.014	0.653 ± 0.024	0.655 ± 0.008	-	0.665 ± 0.007
	MCP	0.655 ± 0.013	0.653 ± 0.008	0.647 ± 0.019	-	0.642 ± 0.027	**0.669 ± 0.016**	0.648 ± 0.012	-	
	$BALC_{JSD}$	-	-	-	0.658 ± 0.003	-	-	-	0.652 ± 0.025	
	$BALC_{KLD}$	-	-	-	0.662± 0.015	-	-	-	0.661 ± 0.014	
0.7	MCD	0.650 ± 0.008	0.640 ± 0.008	**0.658 ± 0.010**	-	0.656 ± 0.008	0.636 ± 0.0134	0.655 ± 0.007	-	0.642 ± 0.015
	MCP	0.653 ± 0.010	0.652 ± 0.009	0.654 ± 0.008	0.646 ± 0.015	-	0.649 ± 0.006	0.651 ± 0.010	-	
	$BALC_{JSD}$	-	-	-	0.642 ± 0.011	-	-	-	0.649 ± 0.008	
	$BALC_{KLD}$	-	-	-	0.653 ± 0.010	-	-	-	0.656 ± 0.012	
0.9	MCD	**0.704 ± 0.009**	0.691 ± 0.012	0.698 ± 0.018	-	0.690 ± 0.022	0.693 ± 0.015	0.692 ± 0.020	-	0.680 ± 0.039
	MCP	0.702 ± 0.019	0.678 ± 0.020	0.700 ± 0.015	-	0.703 ± 0.016	0.680 ± 0.017	0.692 ± 0.009	-	
	$BALC_{JSD}$	-	-	-	0.690 ± 0.006	-	-	-	0.700 ± 0.028	
	$BALC_{KLD}$	-	-	-	0.699 ± 0.011	-	-	-	0.689 ± 0.013	

H.2 PHYSIONET 2015 ECG, $\mathcal{D}_2$

Table 8: Test Set AUC on $\mathcal{D}_2$. Bolded elements represent the best performing method and acquisition function α for each fraction β. No AL represents training without an active learning strategy. Results are averaged across 5 seeds.

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.1	MCD	0.567 ± 0.029	0.591 ± 0.040	0.573 ± 0.063	-	0.547 ± 0.058	0.584 ± 0.055	0.598 ± 0.050	-	0.679 ± 0.040
	MCP	0.567 ± 0.027	0.557 ± 0.032	0.589 ± 0.045	-	0.548 ± 0.036	0.549 ± 0.046	0.554 ± 0.055	-	
	$BALC_{JSD}$	-	-	-	0.576 ± 0.050	-	-	-	0.574 ± 0.057	
	$BALC_{KLD}$	-	-	-	**0.602 ± 0.044**	-	-	-	0.575 ± 0.017	
0.3	MCD	0.675 ± 0.022	0.666 ± 0.053	0.643 ± 0.036	-	0.644 ± 0.019	**0.692 ± 0.020**	0.684 ± 0.035	-	0.605 ± 0.020
	MCP	0.678 ± 0.036	0.660 ± 0.071	0.665 ± 0.051	-	0.643 ± 0.038	0.668 ± 0.020	0.658 ± 0.026	-	
	$BALC_{JSD}$	-	-	-	0.654 ± 0.033	-	-	-	0.677 ± 0.032	
	$BALC_{KLD}$	-	-	-	0.634 ± 0.032	-	-	-	0.672 ± 0.049	
0.5	MCD	0.676 ± 0.0434	0.700 ± 0.031	0.668 ± 0.0185	-	0.709 ± 0.0407	0.694 ± 0.0431	0.669 ± 0.0238	-	0.703 ± 0.032
	MCP	0.687 ± 0.0183	0.695 ± 0.0212	0.712 ± 0.0235	-	0.700 ± 0.0135	0.709 ± 0.0261	0.680 ± 0.0247	-	
	$BALC_{JSD}$	-	-	-	0.701 ± 0.026	-	-	-	0.703 ± 0.018	
	$BALC_{KLD}$	-	-	-	0.705 ± 0.045	-	-	-	**0.726 ± 0.031**	
0.7	MCD	0.758 ± 0.016	0.765 ± 0.027	0.754 ± 0.014	-	0.753 ± 0.020	0.766 ± 0.025	0.755 ± 0.024	-	0.747 ± 0.010
	MCP	0.744 ± 0.031	0.759 ± 0.022	0.745 ± 0.027	-	0.757 ± 0.013	**0.777 ± 0.025**	0.764 ± 0.014	-	
	$BALC_{JSD}$	-	-	-	0.750 ± 0.006	-	-	-	0.746 ± 0.016	
	$BALC_{KLD}$	-	-	-	0.730 ± 0.035	-	-	-	0.761 ± 0.028	
0.9	MCD	0.742 ± 0.016	0.745 ± 0.048	0.757 ± 0.015	-	0.769 ± 0.0261	0.766 ± 0.018	0.754 ± 0.015	-	0.747 ± 0.011
	MCP	0.765 ± 0.013	0.759 ± 0.028	0.751 ± 0.013	-	0.758 ± 0.018	0.759 ± 0.021	0.743 ± 0.025	-	
	$BALC_{JSD}$	-	-	-	0.726 ± 0.008	-	-	-	**0.771 ± 0.018**	
	$BALC_{KLD}$	-	-	-	0.762 ± 0.037	-	-	-	0.749 ± 0.020	

H.3 PHYSIONET 2017 ECG, $\mathcal{D}_3$

Table 9: Test Set AUC on $\mathcal{D}_3$. Bolded elements represent the best performing method and acquisition function α for each fraction β. No AL represents training without an active learning strategy. Results are averaged across 5 seeds.

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.1	MCD	0.628 ± 0.006	0.620 ± 0.006	0.581 ± 0.014	-	0.614 ± 0.03	0.610 ± 0.013	0.562 ± 0.019	-	0.716 ± 0.012
	MCP	0.624 ± 0.017	0.621 ± 0.018	0.623 ± 0.020	-	0.605 ± 0.027	0.613 ± 0.026	0.622 ± 0.026	-	
	BALC_{JSD}	-	-	-	0.613 ± 0.013	-	-	-	0.611 ± 0.015	
	BALC_{KLD}	-	-	-	$\mathbf{0.631 \pm 0.010}$	-	-	-	0.600 ± 0.005	
0.3	MCD	0.705 ± 0.003	0.672 ± 0.009	0.688 ± 0.011	-	0.704 ± 0.016	0.685 ± 0.010	0.684 ± 0.0081	-	0.766 ± 0.012
	MCP	0.688 ± 0.018	0.673 ± 0.007	$\mathbf{0.719 \pm 0.016}$	-	0.671 ± 0.016	0.684 ± 0.018	0.699 ± 0.023	-	
	BALC_{JSD}	-	-	-	0.694 ± 0.006	-	-	-	0.681 ± 0.010	
	BALC_{KLD}	-	-	-	0.703 ± 0.023	-	-	-	0.701 ± 0.015	
0.5	MCD	0.744 ± 0.013	0.735 ± 0.007	0.749 ± 0.012	-	$\mathbf{0.772 \pm 0.015}$	0.743 ± 0.018	0.758 ± 0.009	-	0.790 ± 0.012
	MCP	0.744 ± 0.008	0.733 ± 0.006	0.747 ± 0.004	-	0.741 ± 0.013	0.752 ± 0.019	0.732 ± 0.038	-	
	BALC_{JSD}	-	-	-	0.763 ± 0.022	-	-	-	0.771 ± 0.011	
	BALC_{KLD}	-	-	-	0.769 ± 0.006	-	-	-	0.761 ± 0.003	
0.7	MCD	0.802 ± 0.006	0.811 ± 0.007	0.809 ± 0.004	-	0.807 ± 0.010	0.807 ± 0.003	$\mathbf{0.815 \pm 0.010}$	-	0.810 ± 0.008
	MCP	0.786 ± 0.003	0.782 ± 0.011	0.784 ± 0.016	-	0.772 ± 0.014	0.765 ± 0.013	0.762 ± 0.018	-	
	BALC_{JSD}	-	-	-	0.803 ± 0.011	-	-	-	0.813 ± 0.010	
	BALC_{KLD}	-	-	-	0.809 ± 0.006	-	-	-	0.810 ± 0.005	
0.9	MCD	0.820 ± 0.006	0.824 ± 0.005	0.828 ± 0.004	-	0.821 ± 0.011	0.823 ± 0.005	0.825 ± 0.006	-	0.827 ± 0.004
	MCP	0.826 ± 0.002	0.821 ± 0.007	0.807 ± 0.011	-	0.828 ± 0.008	0.812 ± 0.009	0.808 ± 0.012	-	
	BALC_{JSD}	-	-	-	0.825 ± 0.003	-	-	-	0.824 ± 0.011	
	BALC_{KLD}	-	-	-	0.827 ± 0.005	-	-	-	$\mathbf{0.829 \pm 0.007}$	

H.4 CARDIOLOGY ECG, $\mathcal{D}_4$

Table 10: Test Set AUC on $\mathcal{D}_4$. Bolded elements represent the best performing method and acquisition function α for each fraction β. No AL represents training without an active learning strategy. Results are averaged across 5 seeds.

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.1	MCD	0.475 ± 0.039	$\mathbf{0.518 \pm 0.016}$	0.486 ± 0.011	-	0.485 ± 0.029	0.491 ± 0.022	0.484 ± 0.040	-	0.486 ± 0.023
	MCP	0.508 ± 0.031	0.492 ± 0.022	0.493 ± 0.030	-	0.500 ± 0.024	0.478 ± 0.024	0.492 ± 0.022	-	
	BALC_{JSD}	-	-	-	0.460 ± 0.043	-	-	-	0.487 ± 0.042	
	BALC_{KLD}	-	-	-	0.505 ± 0.032	-	-	-	0.511 ± 0.030	
0.3	MCD	0.487 ± 0.012	0.510 ± 0.018	0.498 ± 0.026	-	0.491 ± 0.014	0.496 ± 0.015	0.500 ± 0.025	-	0.533 ± 0.020
	MCP	0.520 ± 0.007	0.480 ± 0.019	0.494 ± 0.019	-	0.497 ± 0.007	$\mathbf{0.529 \pm 0.035}$	0.498 ± 0.021	-	
	BALC_{JSD}	-	-	-	0.488 ± 0.025	-	-	-	0.487 ± 0.016	
	BALC_{KLD}	-	-	-	0.510 ± 0.030	-	-	-	0.494 ± 0.014	
0.5	MCD	0.563 ± 0.021	$\mathbf{0.591 \pm 0.008}$	0.562 ± 0.011	-	0.557 ± 0.025	0.580 ± 0.006	0.569 ± 0.010	-	0.581 ± 0.019
	MCP	0.529 ± 0.027	0.554 ± 0.024	0.544 ± 0.015	-	0.557 ± 0.021	0.536 ± 0.013	0.526 ± 0.012	-	
	BALC_{JSD}	-	-	-	0.559 ± 0.001	-	-	-	0.559 ± 0.003	
	BALC_{KLD}	-	-	-	0.575 ± 0.028	-	-	-	0.576 ± 0.011	
0.7	MCD	0.637 ± 0.010	0.615 ± 0.010	0.639 ± 0.016	-	0.633 ± 0.016	0.652 ± 0.028	0.662 ± 0.014	-	0.630 ± 0.008
	MCP	0.626 ± 0.018	0.626 ± 0.013	0.623 ± 0.031	-	0.623 ± 0.012	0.623 ± 0.003	0.624 ± 0.010	-	
	BALC_{JSD}	-	-	-	0.634 ± 0.024	-	-	-	$\mathbf{0.648 \pm 0.023}$	
	BALC_{KLD}	-	-	-	0.625 ± 0.015	-	-	-	0.632 ± 0.028	
0.9	MCD	0.651 ± 0.008	0.666 ± 0.011	0.666 ± 0.017	-	0.670 ± 0.007	0.653 ± 0.025	$\mathbf{0.677 \pm 0.009}$	-	0.660 ± 0.013
	MCP	0.655 ± 0.027	0.673 ± 0.009	0.672 ± 0.017	-	0.663 ± 0.006	0.662 ± 0.005	0.670 ± 0.009	-	
	BALC_{JSD}	-	-	-	0.656 ± 0.015	-	-	-	0.656 ± 0.019	
	BALC_{KLD}	-	-	-	0.666 ± 0.025	-	-	-	0.663 ± 0.013	

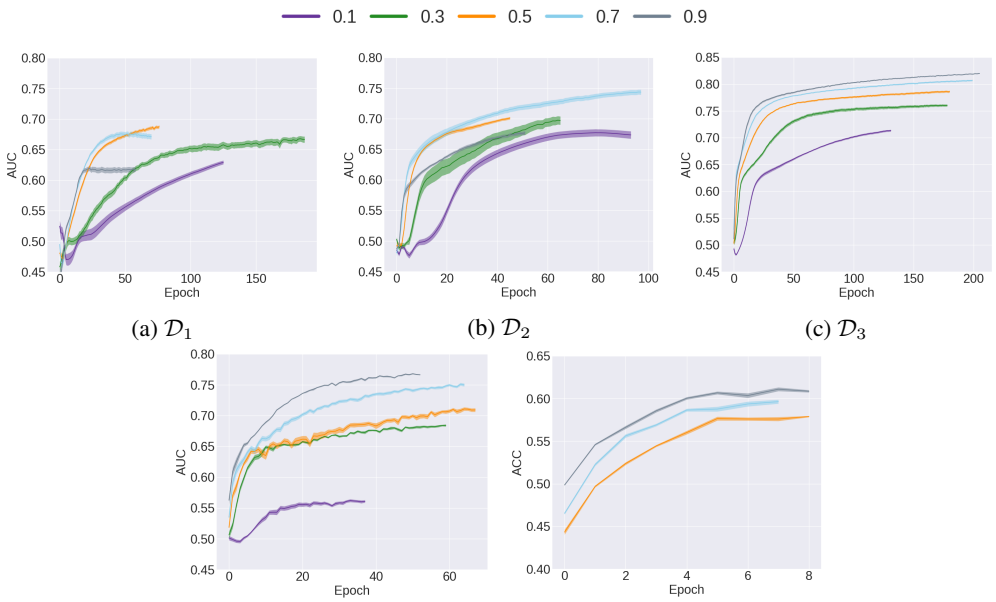
H.5 CIFAR10, $\mathcal{D}_5$

Table 11: Test Set Accuracy on $\mathcal{D}_5$. Bolded elements represent the best performing method and acquisition function α for each fraction β. No AL represents training without an active learning strategy. Results are averaged across 5 seeds.

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.5	MCD	0.566 ± 0.012	0.565 ± 0.010	0.558 ± 0.008	-	0.565 ± 0.008	0.559 ± 0.010	0.562 ± 0.007	-	0.576 ± 0.008
	MCP	0.553 ± 0.009	0.562 ± 0.009	0.561 ± 0.004	-	$\mathbf{0.569 \pm 0.004}$	0.562 ± 0.012	0.554 ± 0.009	-	
	BALC_{JSD}	-	-	-	0.552 ± 0.009	-	-	-	0.565 ± 0.012	
	BALC_{KLD}	-	-	-	0.564 ± 0.010	-	-	-	0.566 ± 0.008	
0.7	MCD	0.586 ± 0.009	0.590 ± 0.009	0.597 ± 0.009	-	0.588 ± 0.006	0.593 ± 0.008	0.594 ± 0.003	-	0.593 ± 0.011
	MCP	$\mathbf{0.600 \pm 0.002}$	0.589 ± 0.010	0.585 ± 0.005	-	0.595 ± 0.010	0.592 ± 0.009	0.599 ± 0.002	-	
	BALC_{JSD}	-	-	-	0.600 ± 0.006	-	-	-	0.589 ± 0.008	
	BALC_{KLD}	-	-	-	0.594 ± 0.007	-	-	-	0.596 ± 0.013	
0.9	MCD	0.618 ± 0.004	0.612 ± 0.007	0.618 ± 0.008	-	0.610 ± 0.004	0.616 ± 0.004	0.615 ± 0.007	-	0.608 ± 0.012
	MCP	0.610 ± 0.002	0.612 ± 0.007	0.610 ± 0.011	-	0.612 ± 0.010	$\mathbf{0.621 \pm 0.004}$	0.608 ± 0.015	-	
	BALC_{JSD}	-	-	-	0.613 ± 0.006	-	-	-	0.609 ± 0.006	
	BALC_{KLD}	-	-	-	0.618 ± 0.010	-	-	-	0.612 ± 0.007	

I Baseline Validation Performance as a Function of Fraction Level, β

The availability of labelled training data is known to affect network performance. To quantify this effect, we illustrate, in Fig. 7, the validation AUC (Accuracy for $\mathcal{D}_5$) for a range of fractions $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$. As expected, we observe a graded response where the larger the amount of labelled training data, the better the generalization performance of the network. This can be seen by the higher AUC achieved when using $\beta = 0.9$ compared to when using $\beta = 0.1$.



(a) $\mathcal{D}_1$ (b) $\mathcal{D}_2$ (c) $\mathcal{D}_3$

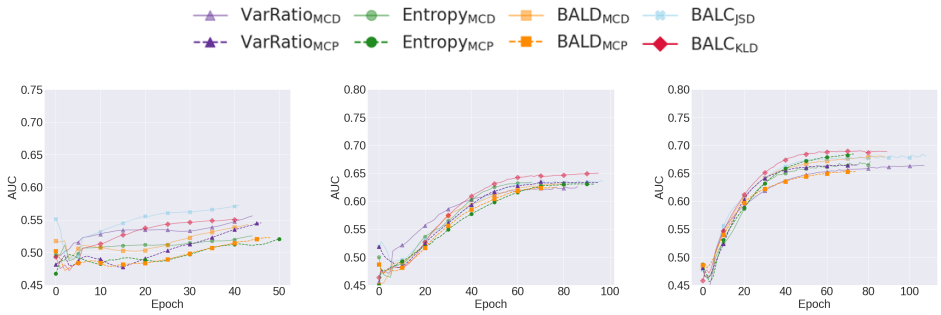
(d) $\mathcal{D}_4$ (e) $\mathcal{D}_5$

Figure 7: Baseline validation AUC for $\mathcal{D}_1$ - $\mathcal{D}_4$ and Accuracy for $\mathcal{D}_5$ at different fraction levels β. These represent the performance curves for the training procedure without active learning. As β increases, performance typically improves.

J Validation Set AUC with Non-Temporal Acquisition Functions in the Absence of Oracle

In the main manuscript, we presented a subset of results for experiments in which oracles are absent and thus unavailable to provide annotations. Instead, unlabelled instances are pseudo-labelled based on network-generated predictions. In this section, we include an exhaustive set of results for all those experiments. More specifically, we illustrate in Figs. 8 - 12 the validation AUC of the various AL methods for datasets $\mathcal{D}_1$ - $\mathcal{D}_5$. At a high level and across datasets, we find that the cold-start problem is likely to occur at low fraction values ($\beta = 0.1$). We include more details in the respective sections.

J.1 PhysioNet 2015 PPG, $\mathcal{D}_1$

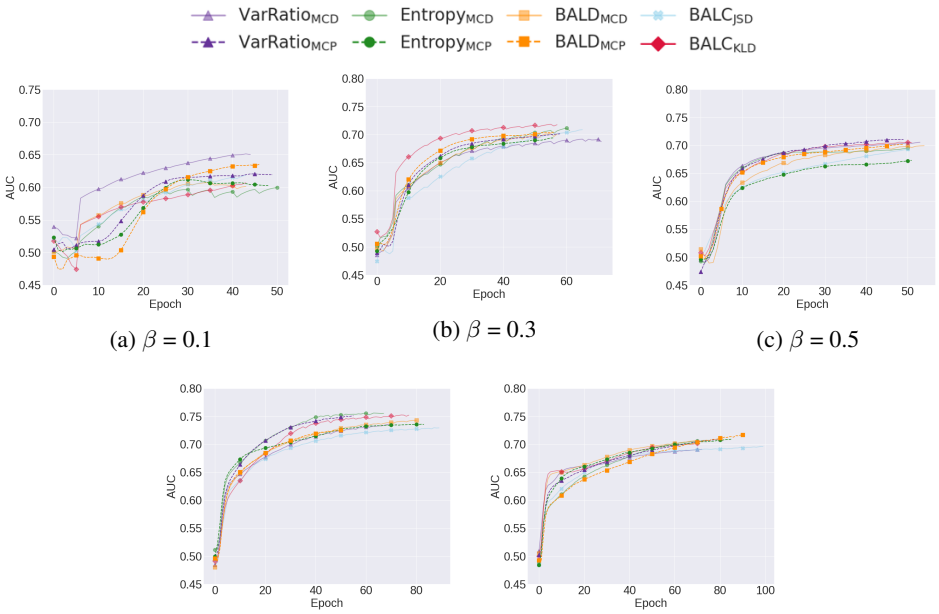


(a) $\beta = 0.1$ (b) $\beta = 0.3$ (c) $\beta = 0.5$

(d) $\beta = 0.7$ (e) $\beta = 0.9$

Figure 8: Mean validation set AUC for the various methodologies and acquisition functions on $\mathcal{D}_1$ at increasing fraction levels $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$. The no-oracle cold-start problem is observed at $\beta = 0.1$ where active learning approaches fail due to few available labelled training instances. Clear benefits of our methods can be seen at $\beta = 0.5, 0.7$. Results are averaged across 5 seeds.

J.2 PHYSIONET 2015 ECG, $\mathcal{D}_2$

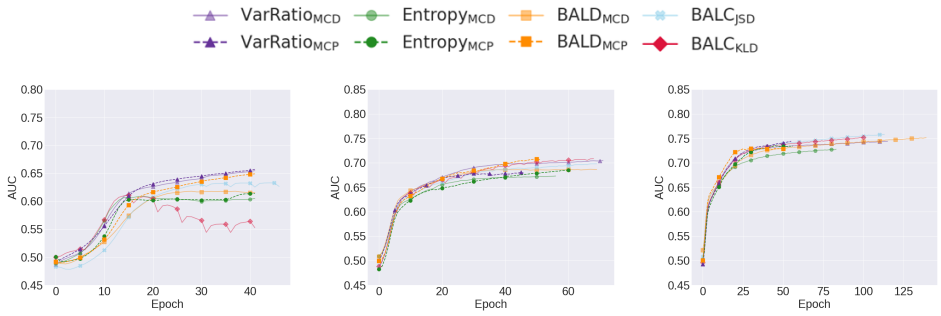


(a) $\beta = 0.1$ (b) $\beta = 0.3$ (c) $\beta = 0.5$

(d) $\beta = 0.7$ (e) $\beta = 0.9$

Figure 9: Mean validation set AUC for the various methodologies and acquisition functions on $\mathcal{D}_2$ at increasing fraction levels $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$. Our methods include MCP and BALC methods. The no-oracle cold-start problem is observed at $\beta = 0.1$ where active learning approaches fail due to few available labelled training instances. However, our approaches outperform all others at $\beta = 0.3, 0.5, 0.7, 0.9$. Results are averaged across 5 seeds.

J.3 PHYSIONET 2017 ECG, $\mathcal{D}_3$

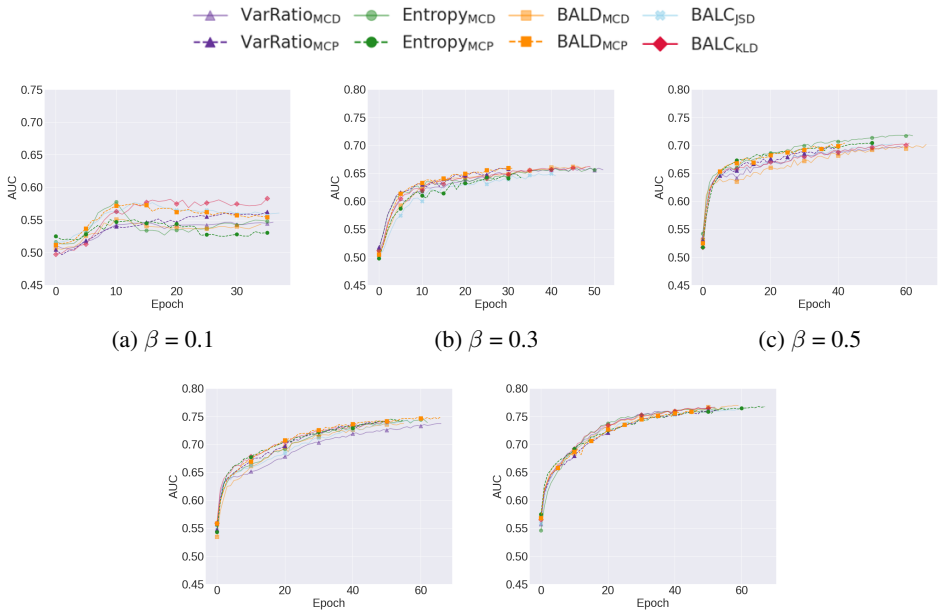


(a) $\beta = 0.1$ (b) $\beta = 0.3$ (c) $\beta = 0.5$

(d) $\beta = 0.7$ (e) $\beta = 0.9$

Figure 10: Mean validation set AUC for the various methodologies and acquisition functions on $\mathcal{D}_3$ at increasing fraction levels $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$. Our methods include MCP and BALC methods. The no-oracle cold-start problem is observed at $\beta = 0.1$ where active learning approaches fail due to few available labelled training instances. Most methods perform on par with the no active learning strategy for this particular dataset. Results are averaged across 5 seeds.

J.4 CARDIOLOGY ECG, $\mathcal{D}_4$

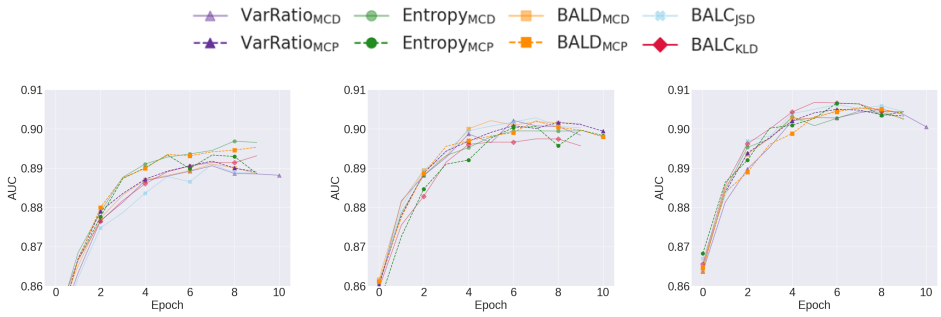


(a) $\beta = 0.1$ (b) $\beta = 0.3$ (c) $\beta = 0.5$

(d) $\beta = 0.7$ (e) $\beta = 0.9$

Figure 11: Mean validation set AUC for the various methodologies and acquisition functions on $\mathcal{D}_4$ at increasing fraction levels $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$. Our methods include MCP and BALC methods. The no-oracle cold-start problem is *not* observed for this dataset. Most methods perform comparably to one another at high values of β. Results are averaged across 5 seeds.

J.5 CIFAR10, $\mathcal{D}_5$

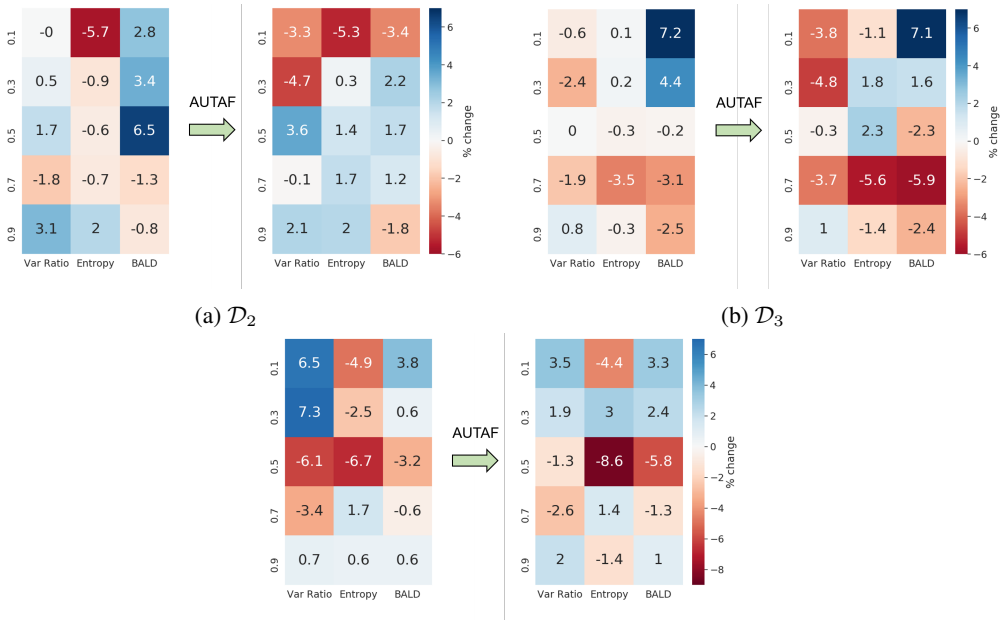


(a) $\beta = 0.5$ (b) $\beta = 0.7$ (c) $\beta = 0.9$

Figure 12: Mean validation set AUC for the various methodologies and acquisition functions on $\mathcal{D}_5$ at increasing fraction levels $\beta = (0.5, 0.7, 0.9)$. Our methods include MCP and BALC methods. At all fractions, we show that MCP methods outperform their MCD counterparts. Results are averaged across 5 seeds.

K Effect of MCP with Tracked Acquisition Functions on Performance

In this section, we are interested in quantifying the effect of implementing a temporal acquisition function in conjunction with MCP on performance. In Fig. 13, we illustrate two columns of matrices. The first column reflects the percent change in generalization performance between implementing MCP and MCD with static temporal functions (i.e., without tracking) for three different datasets. We find that there are mixed results. For example, on $mathcal D_2$ at $\beta = 0.5$, BALD_{MCP} outperforms BALD_{MCD} by 6.5%. However, on $mathcal D_4$ at $\beta = 0.5$, $\text{Entropy}_{\text{MCP}}$ performs worse than $\text{Entropy}_{\text{MCD}}$ by 6.7%. Furthermore, upon applying tracked acquisition functions, we also obtain mixed results. In many cases, there are notable improvements. For example, on $mathcal D_3$ at $\beta = 0.5$, $\text{Temporal Entropy}_{\text{MCP}}$ improves performance by an additional $0.3 + 2.3 = 2.6\%$. On the other hand, at $\beta = 0.7$, $\text{Temporal Entropy}_{\text{MCP}}$ worsens performance by 2.1%. Based on these findings, we would recommend that the utility of temporal acquisition functions be determined on a case-by-case basis.



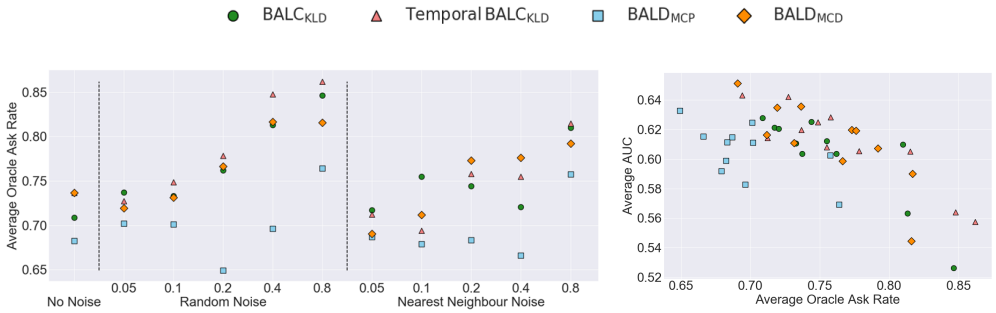
(a) $\mathcal{D}_2$ (b) $\mathcal{D}_3$

(c) $\mathcal{D}_4$

Figure 13: Mean percent change in test AUC when comparing MCP with static and tracked acquisition functions to MCD with their static counterparts on **(a)** $\mathcal{D}_2$ and **(b)** $\mathcal{D}_3$ and **(c)** $\mathcal{D}_4$. We show results for Var Ratio, Entropy, and BALD, at all fractions, $\beta \in [0.1, 0.3, 0.5, 0.7, 0.9]$.

L Degree of Dependence of SoQal on Oracle

A naive argument could claim that SoQal's superiority is simply due to high oracle dependence. To test this hypothesis, we set out to quantify SoQal's dependence on an oracle using the oracle ask-rate: the proportion of all instances acquired whose labels are requested from an oracle. In Fig. 14a, we illustrate this oracle ask-rate for different label noise scenarios.



(a) Oracle ask-rate for different label noise scenarios

(b) Correlation between oracle ask-rate and generalization performance

Figure 14: (a) SoQal's average oracle ask-rate and (b) correlation between oracle ask-rate and average test AUC. Results are averaged across five seeds and all datasets, $\mathcal{D}_1 - \mathcal{D}_5$, and are shown for each acquisition function and label noise scenario.

The oracle ask-rate varies based on the acquisition function used. In Fig. 14a, we show that with 20% random noise, BALD$_{MCP}$ requests labels 65% of the time whereas the remaining acquisition functions do so approximately 77% of the time. We hypothesize that this variability in the oracle ask-rate is due to the variability in the difficulty of the instances acquired by the acquisition functions. In other words, decreased dependence by BALD$_{MCP}$ could be indicative of the acquisition of instances that are relatively farther away from the hyperplane. Thus, they are easier to classify and require less oracle guidance.

In the presence of label noise, decreased oracle dependence is actually associated with improved generalization performance. In Fig. 14b, this is shown by the negative correlation between the oracle ask-rate and the test AUC. Networks are requesting fewer labels and are performing better. Such a finding dispels the original claim and reaffirms the conclusion that SoQal knows *when* to request a label from an oracle.

M Effect of Data Availability on Oracle Ask-Rate

As we performed all experiments on small, labelled datasets, we wanted to investigate the role of data availability on the amount of independence an algorithm can withstand. We hypothesize that access to large, labelled datasets would allow SoQal to decrease its overall dependence on an oracle while maintaining its generalization performance. In Table 12, we illustrate the effect of increasing the amount of labelled training data N-Fold, where $N = [3, 5, 7, 9]$, on the oracle ask-rate (OAR) and the generalization performance of a network.

Increasing the amount of labelled training data can drastically decrease oracle dependence *without* significantly affecting generalization performance. In Table 12, we show that a 5-fold increase in sample-size results in $\mathrm{OAR} = 100\% \rightarrow 59\%$, a 41% reduction in dependence while maintaining the test $\mathrm{AUC} \approx 0.59$. In other words, 41% of a physician's time can be reliably saved.

Table 12: Mean test AUC and oracle ask rate of a 100% oracle strategy and SoQal in response to more labelled training data. Results are shown for $\mathcal{D}_4$ and $\mathrm{BALD}_{\mathrm{MCD}}$ at $S = 0.15$ across five seeds. Original represents the small sample size used for all previous experiments.

Oracle Questioning Method	100% Oracle	SoQal				
Sample Size Factor	Original	Original	3-Fold	5-Fold	7-Fold	9-Fold
Average OAR %	100	48	56	59	68	67
AUC	0.585	0.468	0.507	0.594	0.659	0.657

N CONTROLLING ORACLE DEPENDENCE VIA HELLINGER THRESHOLD, S

In this section, we set out to investigate the degree to which the Hellinger threshold, S, acts as a knob that controls the oracle ask-rate (OAR). We expect this behaviour to arise from our design that when $\mathcal{D}_H < S$, all label requests are sent to the oracle. In Table 13, we show how different values of S affect the oracle ask-rate and the test AUC.

There exists a positive relationship between S and the oracle ask-rate. As $S = 0.100 \rightarrow 0.400$, the $\text{OAR} = 86\% \rightarrow 100\%$. In the presence of a noise-free oracle, we would expect this increased dependence to be associated with improved generalization performance. We observe such behaviour as $\text{OAR} = 86\% \rightarrow 94\%$ and $\text{AUC} = 0.716 \rightarrow 0.768$. However, increased dependence beyond this point is a detriment to performance. This finding reaffirms our previous hypothesis that the original labels in the dataset may be noisy. Therefore, a sub-100% OAR scenario in which these particular noisy labels are not requested from the oracle would be advantageous.

Table 13: Mean test AUC of SoQal and oracle ask rate in response to various threshold values, S. Results are shown for $\mathcal{D}_3$ and BALD_{MCD} across five seeds. Experiments are performed with a noise-free oracle.

Threshold, S	0.100	0.125	0.150	0.175	0.200	0.300	0.400
Average OAR %	86	85	89	90	94	100	100
AUC	0.716	0.744	0.721	0.753	0.768	0.743	0.755

O Performance of Oracle Selection Strategies with Noisy Oracle

Over-reliance on an oracle could be detrimental for an active learning algorithm if that oracle is unable to label instances accurately. In the case of physicians, this inability could arise due to poor training, fatigue, or the difficulty of a particular case being diagnosed. We simulate these scenarios by injecting label noise of various magnitude into the datasets. In this section, we illustrate the performance of three oracle selection strategies, SoQal, Epsilon Greedy, and Entropy Response, in response to label noise. The results are shown for random and nearest neighbour noise in Secs. O.1 and O.2, respectively.

O.1 Label Noise - Random

Although random noise can be considered an extreme case, it is nonetheless plausible in certain scenarios where labellers are poorly trained or the task at hand is too difficult. In this section, we illustrate, in Tables 14a - 14c, the degree to which the test AUC is affected by the introduction of random label noise during the active learning procedure.

As expected, extreme levels of noise negatively affect performance. For instance, this can be seen in Table 14a at $\mathcal{D}_2$ using BALD_{MCD} where increasing the level of random noise from $5\% \rightarrow 80\%$ leads to a reduction of $\text{AUC} = 0.679 \rightarrow 0.556$. Across most noise levels, SoQal continues to outperform Epsilon Greedy and Entropy Response. This finding is consistent with that presented in the main manuscript and illustrates the relative robustness of SoQal to label noise.

Table 14: Mean test AUC of oracle questioning strategies as a function of increasing levels of *random* label noise by the oracle. Results are shown for datasets $\mathcal{D}_1 - \mathcal{D}_5$ and all acquisition functions. Mean and standard deviation values are shown across five seeds.

(a) SoQal

Dataset	Ac. Function α	Random Noise Level				
		0.05	0.10	0.20	0.40	0.80
$\mathcal{D}_1$	BALD_{MCD}	0.595 ± 0.053	0.554 ± 0.028	0.558 ± 0.042	0.600 ± 0.029	0.511 ± 0.055
	BALD_{MCP}	0.659 ± 0.014	0.650 ± 0.027	0.636 ± 0.029	0.549 ± 0.058	0.528 ± 0.029
	BALC_{KLD}	0.564 ± 0.058	0.570 ± 0.045	0.562 ± 0.067	0.498 ± 0.038	0.477 ± 0.011
	Temporal BALC_{KLD}	0.634 ± 0.026	0.597 ± 0.035	0.611 ± 0.040	0.494 ± 0.034	0.490 ± 0.026
$\mathcal{D}_2$	BALD_{MCD}	0.679 ± 0.017	0.659 ± 0.042	0.646 ± 0.044	0.602 ± 0.047	0.556 ± 0.065
	BALD_{MCP}	0.643 ± 0.020	0.677 ± 0.053	0.637 ± 0.042	0.619 ± 0.033	0.602 ± 0.041
	BALC_{KLD}	0.652 ± 0.037	0.659 ± 0.056	0.649 ± 0.054	0.614 ± 0.016	0.522 ± 0.032
	Temporal BALC_{KLD}	0.655 ± 0.048	0.701 ± 0.029	0.628 ± 0.074	0.594 ± 0.041	0.581 ± 0.032
$\mathcal{D}_3$	BALD_{MCD}	0.750 ± 0.017	0.742 ± 0.031	0.718 ± 0.037	0.646 ± 0.023	0.584 ± 0.017
	BALD_{MCP}	0.724 ± 0.022	0.707 ± 0.021	0.682 ± 0.038	0.629 ± 0.029	0.537 ± 0.025
	BALC_{KLD}	0.724 ± 0.032	0.725 ± 0.028	0.702 ± 0.024	0.651 ± 0.046	0.564 ± 0.040
	Temporal BALC_{KLD}	0.725 ± 0.031	0.738 ± 0.013	0.705 ± 0.017	0.596 ± 0.071	0.546 ± 0.041
$\mathcal{D}_4$	BALD_{MCD}	0.506 ± 0.019	0.479 ± 0.022	0.496 ± 0.023	0.490 ± 0.010	0.518 ± 0.029
	BALD_{MCP}	0.499 ± 0.037	0.508 ± 0.022	0.523 ± 0.027	0.495 ± 0.023	0.503 ± 0.021
	BALC_{KLD}	0.491 ± 0.026	0.481 ± 0.023	0.496 ± 0.031	0.518 ± 0.012	0.525 ± 0.011
	Temporal BALC_{KLD}	0.522 ± 0.016	0.505 ± 0.027	0.501 ± 0.021	0.511 ± 0.025	0.515 ± 0.031

(b) Epsilon Greedy

Dataset	Ac. Function α	Random Noise Level 0.05	0.10	0.20	0.40	0.80
$\mathcal{D}_1$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.496± 0.058	0.494 ± 0.029	0.476 ± 0.030	0.507 ± 0.044	0.501 ± 0.056
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.557 ± 0.018	0.549 ± 0.036	0.508 ± 0.032	0.511 ± 0.033	0.497 ± 0.057
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.517 ± 0.014	0.518 ± 0.035	0.504 ± 0.028	0.506 ± 0.034	0.498 ± 0.017
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.525 ± 0.037	0.512 ± 0.040	0.501 ± 0.025	0.493 ± 0.019	0.497 ± 0.039
$\mathcal{D}_2$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.600± 0.053	0.628 ± 0.053	0.589 ± 0.037	0.612 ± 0.022	0.555 ± 0.041
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.629 ± 0.023	0.614 ± 0.048	0.536 ± 0.081	0.575 ± 0.029	0.588 ± 0.050
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.619 ± 0.038	0.586 ± 0.054	0.629 ± 0.061	0.613 ± 0.045	0.582 ± 0.067
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.630 ± 0.041	0.652 ± 0.029	0.579 ± 0.034	0.610 ± 0.036	0.564 ± 0.035
$\mathcal{D}_3$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.663 ± 0.018	0.661 ± 0.011	0.632 ± 0.021	0.626± 0.012	0.588 ± 0.017
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.671 ± 0.017	0.670 ± 0.016	0.639 ± 0.024	0.623 ± 0.019	0.574 ± 0.041
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.665 ± 0.022	0.650 ± 0.014	0.664 ± 0.013	0.618 ± 0.034	0.595 ± 0.031
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.661 ± 0.012	0.651 ± 0.018	0.651 ± 0.016	0.629 ± 0.019	0.612 ± 0.049
$\mathcal{D}_4$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.473 ± 0.030	0.480 ± 0.033	0.469 ± 0.024	0.468± 0.018	0.493 ± 0.015
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.508 ± 0.016	0.495 ± 0.019	0.498 ± 0.043	0.494 ± 0.032	0.497 ± 0.015
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.492 ± 0.026	0.496 ± 0.021	0.481 ± 0.025	0.491 ± 0.020	0.498 ± 0.021
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.514 ± 0.017	0.528 ± 0.017	0.500 ± 0.008	0.498 ± 0.033	0.504 ± 0.037

(c) Entropy Response

Dataset	Ac. Function α	Random Noise Level 0.05	0.10	0.20	0.40	0.80
$\mathcal{D}_1$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.495 ± 0.038	0.497 ± 0.057	0.498 ± 0.057	0.486 ± 0.044	0.512 ± 0.057
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.534 ± 0.018	0.584 ± 0.073	0.565 ± 0.033	0.619 ± 0.022	0.518 ± 0.028
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.535 ± 0.038	0.521 ± 0.042	0.514 ± 0.053	0.511 ± 0.027	0.525 ± 0.037
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.526 ± 0.040	0.538 ± 0.036	0.504 ± 0.028	0.501 ± 0.036	0.500 ± 0.004
$\mathcal{D}_2$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.587 ± 0.044	0.564 ± 0.058	0.586 ± 0.047	0.613 ± 0.083	0.551 ± 0.031
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.624 ± 0.044	0.598 ± 0.057	0.573 ± 0.053	0.560 ± 0.081	0.530 ± 0.015
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.616 ± 0.043	0.653 ± 0.049	0.624 ± 0.051	0.565 ± 0.055	0.579 ± 0.019
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.635 ± 0.045	0.603 ± 0.050	0.590 ± 0.042	0.602 ± 0.046	0.579 ± 0.041
$\mathcal{D}_3$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.592 ± 0.015	0.604 ± 0.017	0.603 ± 0.016	0.603 ± 0.016	0.605 ± 0.018
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.694 ± 0.047	0.730 ± 0.029	0.666 ± 0.034	0.639 ± 0.031	0.599 ± 0.035
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.631 ± 0.006	0.631 ± 0.009	0.622 ± 0.011	0.631 ± 0.025	0.564 ± 0.046
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.602 ± 0.011	0.622 ± 0.018	0.630 ± 0.014	0.618 ± 0.040	0.565 ± 0.050
$\mathcal{D}_4$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.472 ± 0.029	0.472 ± 0.030	0.486 ± 0.008	0.476 ± 0.038	0.481 ± 0.038
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.511 ± 0.021	0.510 ± 0.023	0.525 ± 0.033	0.498 ± 0.041	0.497 ± 0.017
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.468 ± 0.022	0.472 ± 0.029	0.477 ± 0.029	0.483 ± 0.018	0.475 ± 0.032
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.482 ± 0.023	0.491 ± 0.013	0.490 ± 0.021	0.487 ± 0.031	0.515 ± 0.022

O.2 LABEL NOISE - NEAREST NEIGHBOUR

Nearest neighbour noise is more realistic than that which is random as it may simulate uncertainty in diagnoses made by physicians. In this section, we illustrate, in Tables 15a - 15c, the degree to which the test AUC is affected by the introduction of nearest neighbour label noise during the active learning procedure.

As expected, extreme levels of noise negatively affect performance. For instance, this can be seen in Table 15a at $\mathcal{D}_3$ using BALD_{MCD} where increasing the level of nearest neighbour noise from $5\% \rightarrow 80\%$ leads to a reduction of the $\text{AUC} = 0.744 \rightarrow 0.694$. SoQal continues to outperform Epsilon Greedy and Entropy Response across most of the noise levels. Building on the previous example, with 80% nearest neighbour noise, SoQal achieves an $\text{AUC} = 0.694$ whereas Epsilon Greedy and Entropy Response achieve an $\text{AUC} = 0.632$ and 0.587, respectively. Such a finding is similar to that arrived at with Random Noise and implies that SoQal is relatively more robust to noisy oracles than these other methods.

Table 15: Mean test AUC of oracle questioning strategies as a function of increasing levels of *nearest neighbour* label noise by the oracle. Results are shown for datasets $\mathcal{D}_1 - \mathcal{D}_5$ and all acquisition functions. Mean and standard deviation values are shown across five seeds.

(a) SoQal

Dataset	Ac. Function α	Nearest Neighbour Noise Level				
		0.05	0.10	0.20	0.40	0.80
$\mathcal{D}_1$	BALD_{MCD}	0.614 ± 0.043	0.571 ± 0.037	0.618 ± 0.015	0.557 ± 0.042	0.540 ± 0.052
	BALD_{MCP}	0.633 ± 0.011	0.617 ± 0.095	0.641 ± 0.023	0.632 ± 0.026	0.591 ± 0.047
	BALC_{KLD}	0.628 ± 0.049	0.586 ± 0.032	0.616 ± 0.024	0.604 ± 0.022	0.558 ± 0.069
	Temporal BALC_{KLD}	0.557 ± 0.045	0.647 ± 0.060	0.620 ± 0.036	0.625 ± 0.038	0.577 ± 0.039
$\mathcal{D}_2$	BALD_{MCD}	0.694 ± 0.022	0.631 ± 0.020	0.682 ± 0.036	0.658 ± 0.038	0.647 ± 0.039
	BALD_{MCP}	0.605 ± 0.054	0.660 ± 0.067	0.656 ± 0.029	0.618 ± 0.058	0.605 ± 0.081
	BALC_{KLD}	0.655 ± 0.015	0.660 ± 0.037	0.671 ± 0.078	0.649 ± 0.024	0.678 ± 0.023
	Temporal BALC_{KLD}	0.702 ± 0.044	0.654 ± 0.024	0.686 ± 0.038	0.638 ± 0.042	0.631 ± 0.020
$\mathcal{D}_3$	BALD_{MCD}	0.744 ± 0.023	0.745 ± 0.021	0.709 ± 0.028	0.700 ± 0.026	0.694 ± 0.014
	BALD_{MCP}	0.706 ± 0.029	0.736 ± 0.036	0.727 ± 0.023	0.712 ± 0.018	0.682 ± 0.017
	BALC_{KLD}	0.718 ± 0.029	0.729 ± 0.028	0.735 ± 0.021	0.680 ± 0.050	0.688 ± 0.009
	Temporal BALC_{KLD}	0.727 ± 0.033	0.725 ± 0.033	0.724 ± 0.018	0.700 ± 0.022	0.645 ± 0.062
$\mathcal{D}_4$	BALD_{MCD}	0.517 ± 0.034	0.477 ± 0.027	0.493 ± 0.034	0.498 ± 0.036	0.459 ± 0.035
	BALD_{MCP}	0.492 ± 0.027	0.491 ± 0.027	0.502 ± 0.036	0.532 ± 0.040	0.507 ± 0.042
	BALC_{KLD}	0.494 ± 0.024	0.494 ± 0.016	0.504 ± 0.026	0.506 ± 0.031	0.503 ± 0.021
	Temporal BALC_{KLD}	0.504 ± 0.018	0.515 ± 0.013	0.529 ± 0.027	0.507 ± 0.014	0.508 ± 0.026

(b) Epsilon Greedy

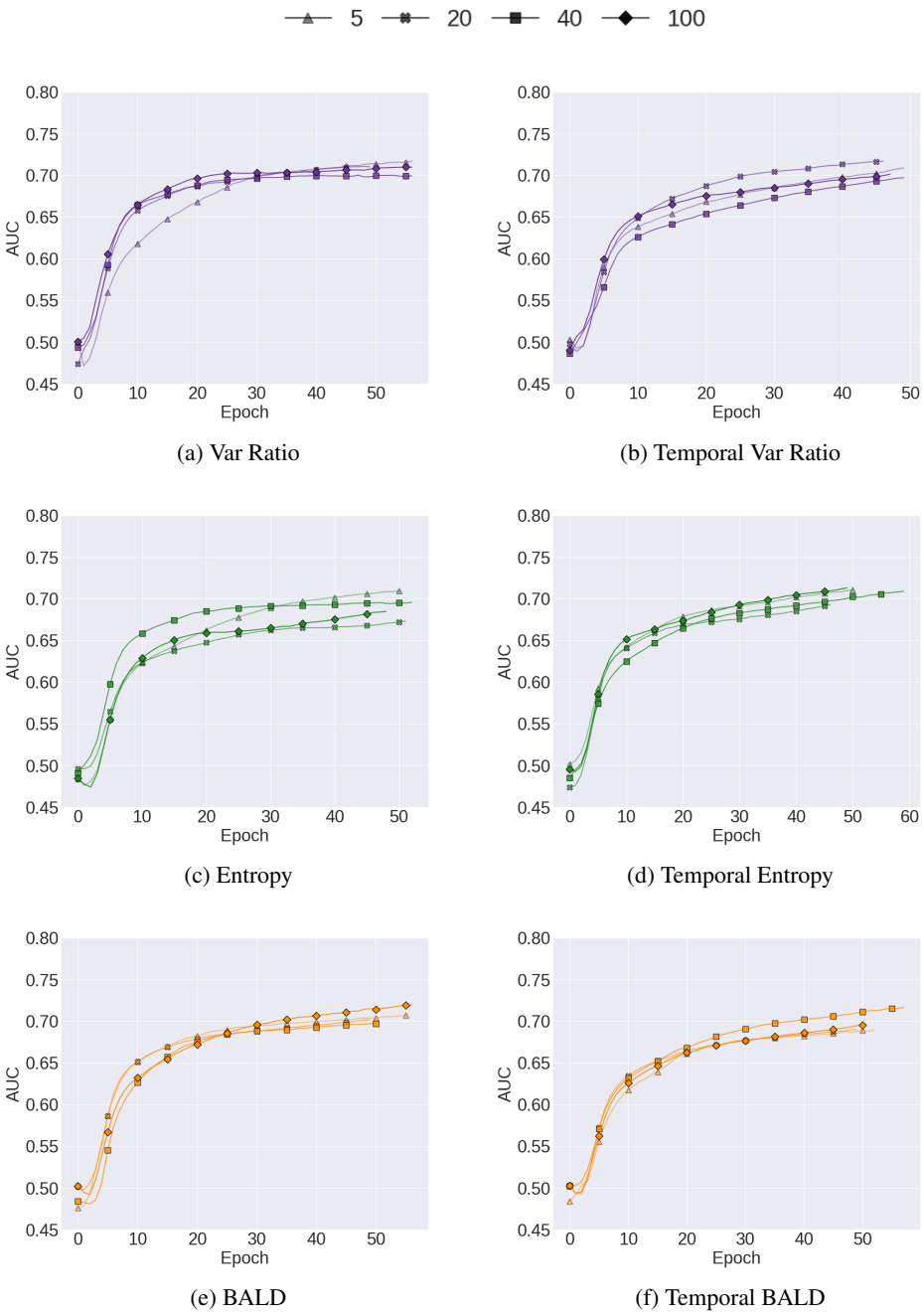
Dataset	Ac. Function α	Nearest Neighbour Noise Level 0.05	0.10	0.20	0.40	0.80
$\mathcal{D}_1$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.503 ± 0.040	0.480 ± 0.023	0.514 ± 0.050	0.481 ± 0.038	0.456 ± 0.031
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.501 ± 0.014	0.508 ± 0.036	0.522 ± 0.052	0.482 ± 0.023	0.496 ± 0.041
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.541 ± 0.035	0.503 ± 0.033	0.486 ± 0.047	0.500 ± 0.041	0.473 ± 0.026
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.516 ± 0.024	0.523 ± 0.044	0.495 ± 0.046	0.491 ± 0.013	0.483 ± 0.041
$\mathcal{D}_2$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.584 ± 0.066	0.610 ± 0.042	0.597 ± 0.036	0.616 ± 0.054	0.593 ± 0.054
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.565 ± 0.031	0.589 ± 0.075	0.616 ± 0.059	0.605 ± 0.047	0.586 ± 0.047
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.608 ± 0.031	0.607 ± 0.040	0.590 ± 0.055	0.538 ± 0.037	0.585 ± 0.053
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.647 ± 0.044	0.591 ± 0.033	0.640 ± 0.044	0.576 ± 0.031	0.589 ± 0.030
$\mathcal{D}_3$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.656 ± 0.021	0.655 ± 0.014	0.665 ± 0.010	0.643 ± 0.021	0.632 ± 0.010
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.660 ± 0.022	0.657 ± 0.023	0.659 ± 0.003	0.664 ± 0.023	0.634 ± 0.013
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.608 ± 0.031	0.607 ± 0.040	0.590 ± 0.055	0.538 ± 0.037	0.585 ± 0.053
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.644 ± 0.016	0.651 ± 0.011	0.658 ± 0.013	0.634 ± 0.016	0.627 ± 0.014
$\mathcal{D}_4$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.438 ± 0.014	0.457 ± 0.022	0.442 ± 0.018	0.456 ± 0.028	0.428 ± 0.024
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.489 ± 0.018	0.489 ± 0.021	0.474 ± 0.023	0.485 ± 0.019	0.486 ± 0.015
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.485 ± 0.029	0.487 ± 0.019	0.495 ± 0.023	0.488 ± 0.028	0.481 ± 0.021
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.486 ± 0.018	0.500 ± 0.029	0.486 ± 0.027	0.468 ± 0.017	0.487 ± 0.028

(c) Entropy Response

Dataset	Ac. Function α	Nearest Neighbour Noise Level 0.05	0.10	0.20	0.40	0.80
$\mathcal{D}_1$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.494 ± 0.037	0.474 ± 0.027	0.492 ± 0.051	0.482 ± 0.033	0.444 ± 0.006
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.511 ± 0.019	0.562 ± 0.052	0.509 ± 0.042	0.572 ± 0.060	0.495 ± 0.041
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.513 ± 0.020	0.517 ± 0.035	0.504 ± 0.034	0.498 ± 0.023	0.487 ± 0.023
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.500 ± 0.043	0.540 ± 0.025	0.503 ± 0.043	0.516 ± 0.024	0.490 ± 0.024
$\mathcal{D}_2$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.585 ± 0.045	0.630 ± 0.056	0.600 ± 0.045	0.585 ± 0.046	0.586 ± 0.063
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.633 ± 0.060	0.626 ± 0.064	0.618 ± 0.055	0.647 ± 0.077	0.619 ± 0.055
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.605 ± 0.049	0.572 ± 0.032	0.630 ± 0.081	0.581 ± 0.031	0.589 ± 0.061
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.625 ± 0.030	0.599 ± 0.024	0.613 ± 0.050	0.614 ± 0.052	0.606 ± 0.054
$\mathcal{D}_3$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.604 ± 0.017	0.589 ± 0.013	0.592 ± 0.014	0.592 ± 0.014	0.587 ± 0.012
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.636 ± 0.030	0.635 ± 0.030	0.640 ± 0.040	0.634 ± 0.039	0.623 ± 0.032
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.632 ± 0.008	0.633 ± 0.008	0.630 ± 0.005	0.629 ± 0.004	0.625 ± 0.008
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.631 ± 0.013	0.630 ± 0.013	0.637 ± 0.013	0.630 ± 0.014	0.629 ± 0.009
$\mathcal{D}_4$	$\mathrm{BALD}_{\mathrm{MCD}}$	0.475 ± 0.035	0.493 ± 0.025	0.471 ± 0.031	0.468 ± 0.027	0.481 ± 0.035
	$\mathrm{BALD}_{\mathrm{MCP}}$	0.508 ± 0.024	0.512 ± 0.020	0.513 ± 0.019	0.499 ± 0.012	0.492 ± 0.016
	$\mathrm{BALC}_{\mathrm{KLD}}$	0.483 ± 0.031	0.476 ± 0.033	0.473 ± 0.026	0.479 ± 0.021	0.479 ± 0.032
	Temporal $\mathrm{BALC}_{\mathrm{KLD}}$	0.490 ± 0.012	0.497 ± 0.030	0.466 ± 0.013	0.485 ± 0.016	0.500 ± 0.013

P Effect of Number of Monte Carlo Samples, T, on Performance

The number of MC samples, T, within an AL framework can be associated with an improved approximation of the version space. This, in turn, should lead to improved AL results. To quantify the effect of the number of MC samples on performance, we illustrate in Fig. 15, the validation AUC for experiments conducted with $T = (5, 20, 40, 100)$. We show that there does not exist a simple proportional relationship between the number of MC samples and performance. This can be seen by the relatively strong generalization performance of models when $T = 100$ in Figs. 15c, 15h, and 15i and poorer performance when $T = 100$. This suggests that our family of methods can perform well without being computationally expensive.



(a) Var Ratio

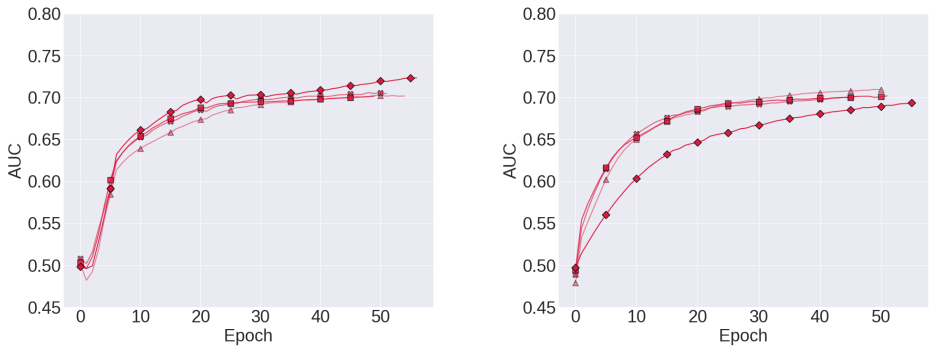
(b) Temporal Var Ratio

(c) Entropy

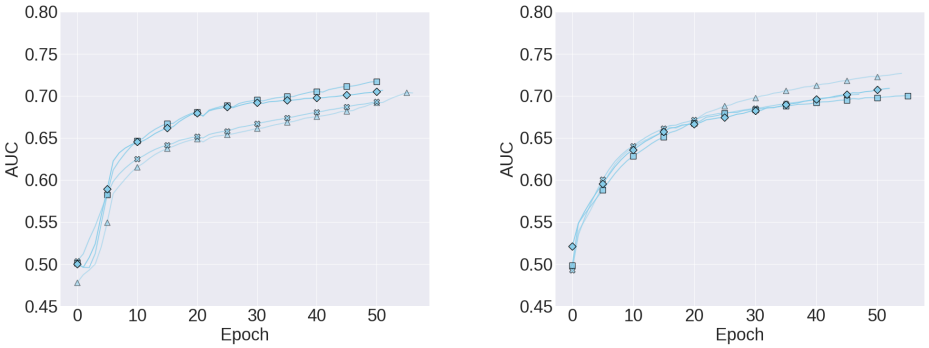
(d) Temporal Entropy

(e) BALD

(f) Temporal BALD



(g) BALC_{KLD} (h) Temporal BALC_{KLD}

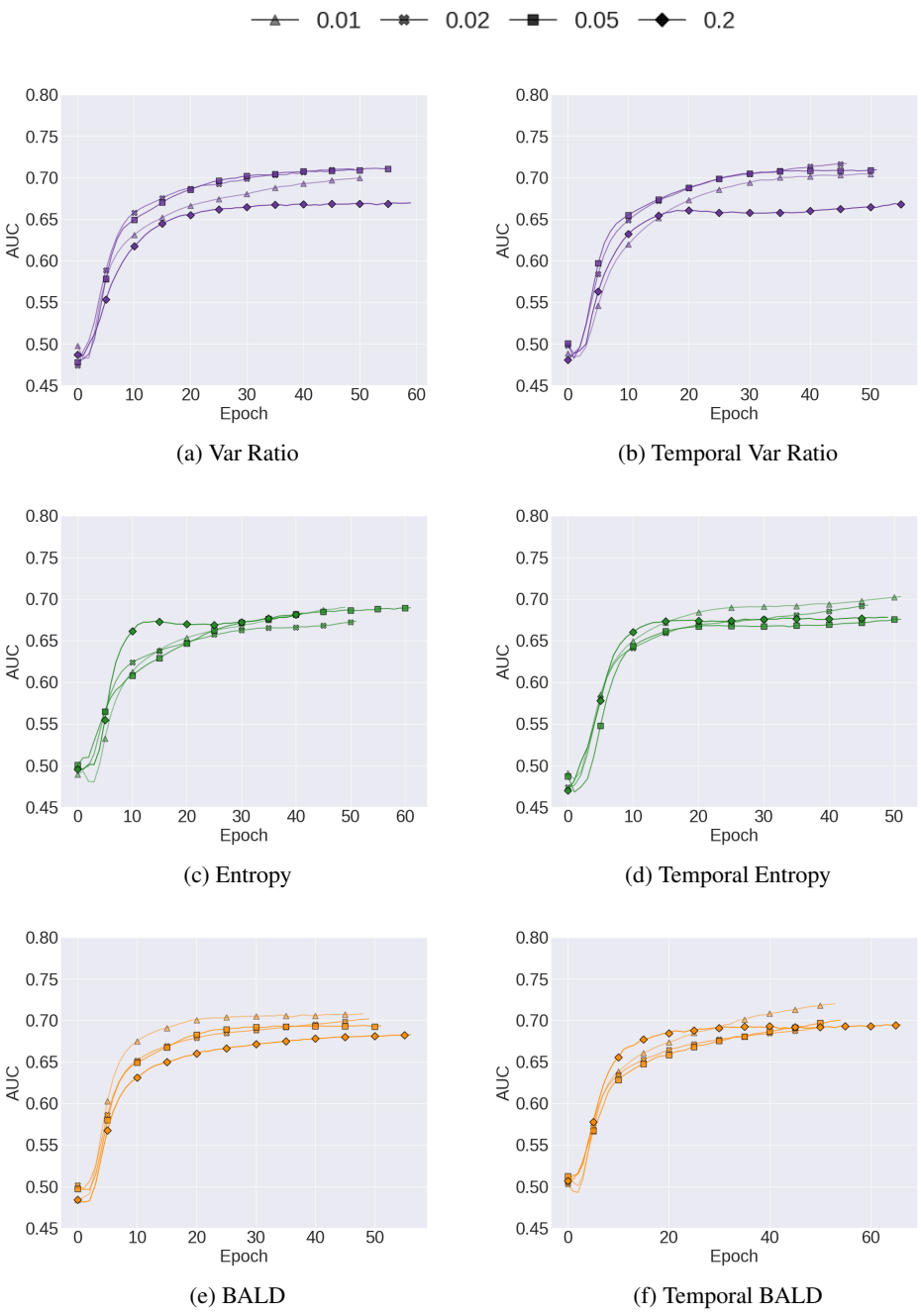


(i) BALC_{JSD} (j) Temporal BALC_{JSD}

Figure 15: Mean validation AUC as a function of number of Monte Carlo samples T for the different acquisition functions using the MCP method. The acquisition percentage and acquisition epochs were fixed at $b = 2\%$ and $\tau = 5$, respectively. These experiments are performed on $\mathcal{D}_2$ at a fraction of $\beta = 0.5$. Results are averaged across 5 seeds.

Q EFFECT OF ACQUISITION PERCENTAGE, b, ON PERFORMANCE

The number of unlabelled instances acquired during the AL procedure can have a strong effect on the generalization performance of networks. We investigate the effect of this on our family of methods and illustrate the results in Fig. 16 when conducting experiments for $b = (1\%, 2\%, 5\%, 20\%)$. Contrary to expectations that more acquisition is better, we show that acquiring large amounts of data is actually detrimental. This can be seen by the poorer performance attributed to $b = 20\%$ in, for instance, Figs. 16b, 16f, and 16g. We hypothesize that this is due to larger magnitude 1) distribution shifts and 2) label noise brought about by the absence of an oracle.



(a) Var Ratio

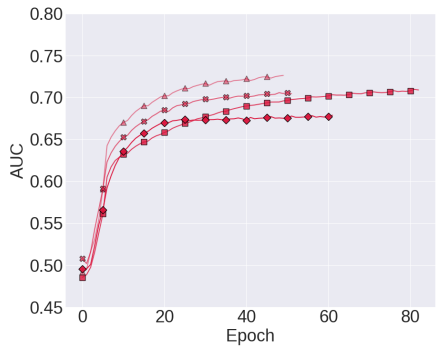
(b) Temporal Var Ratio

(c) Entropy

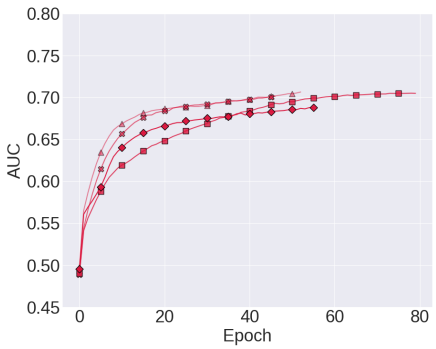
(d) Temporal Entropy

(e) BALD

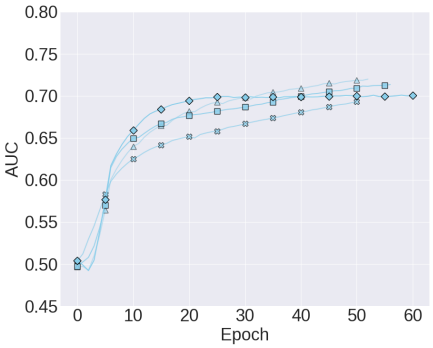
(f) Temporal BALD



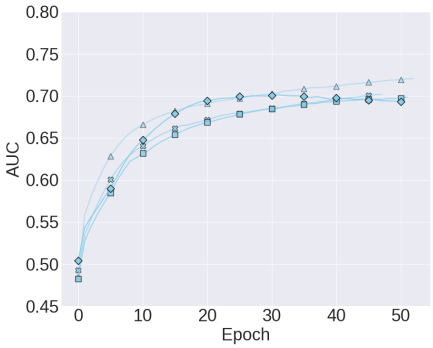
(g) BALC_{KLD}



(h) Temporal BALC_{KLD}



(i) BALC_{JSD}

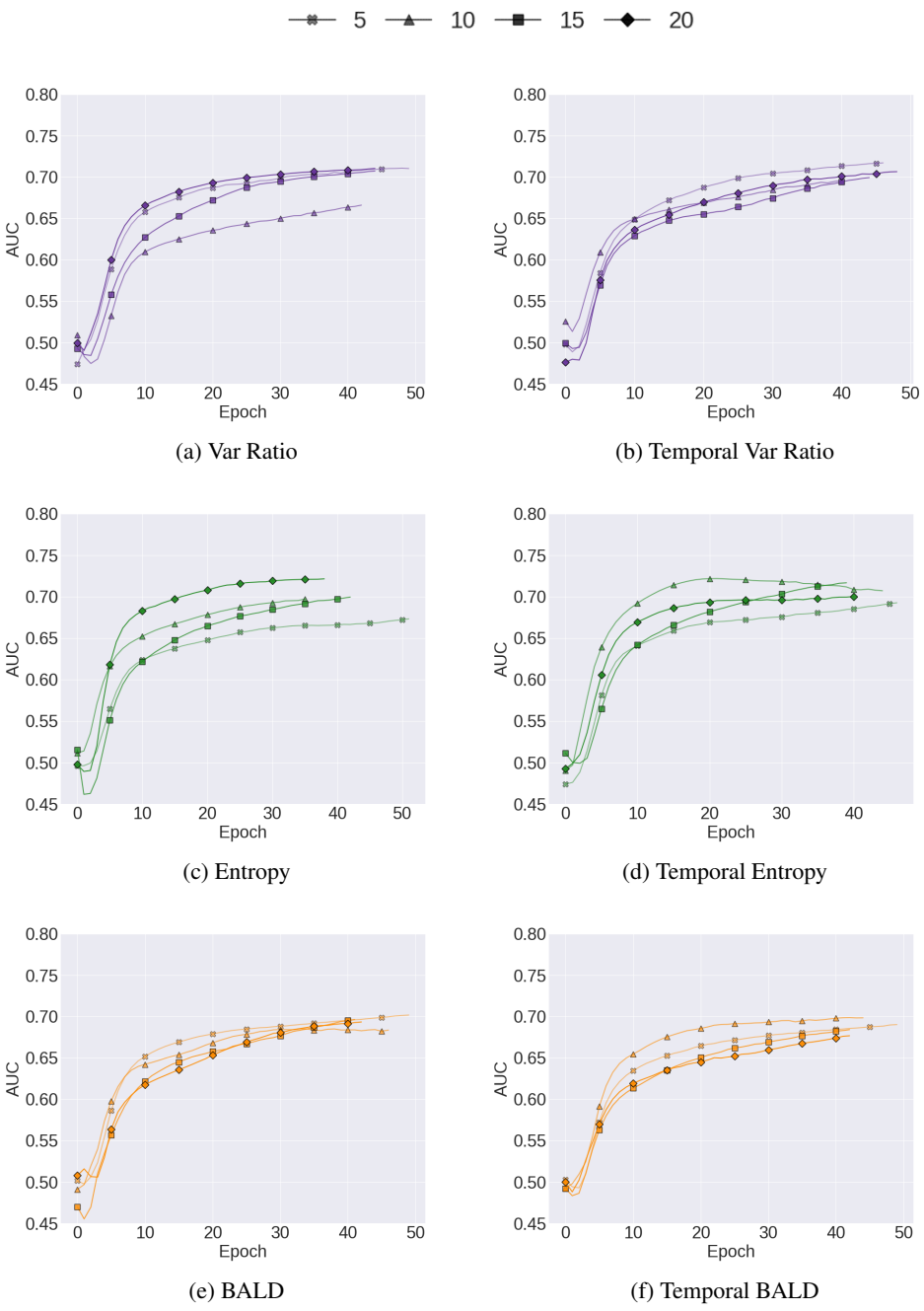


(j) Temporal BALC_{JSD}

Figure 16: Mean AUC of the validation set as a function of acquisition percentage b for the different acquisition functions using the MCP method. These experiments are performed on $\mathcal{D}_2$ at a fraction $\beta = 0.5$. MC samples and acquisition epochs were fixed at $T = 20$ and $\tau = 5$, respectively. Results are averaged across 5 seeds.

R Effect of Acquisition Epochs, τ, on Performance

As outlined in the main manuscript, the control vs. shock trade-off must be balanced to ensure good generalization performance of an AL procedure. Acquiring instances too early and frequently can lead to instabilities in the training procedure. Conversely, inadequate sampling of unlabelled instances starves the network of much needed data. To quantify this trade-off, we illustrate in Fig. 17, the performance of our family of methods when $\tau = (5, 10, 15, 20)$. Although one value that guarantees best performance for all experiments does not exist, $\tau = 10$ or $\tau = 15$ seem to outperform the others, on average.



(a) Var Ratio

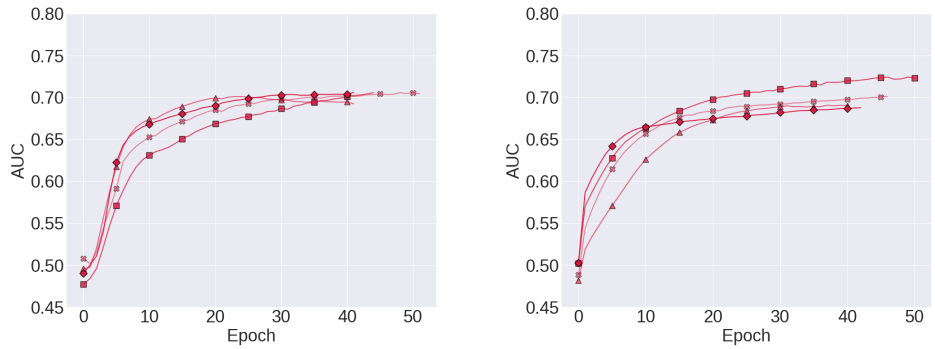
(b) Temporal Var Ratio

(c) Entropy

(d) Temporal Entropy

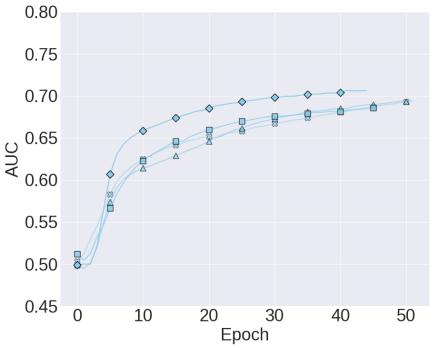
(e) BALD

(f) Temporal BALD

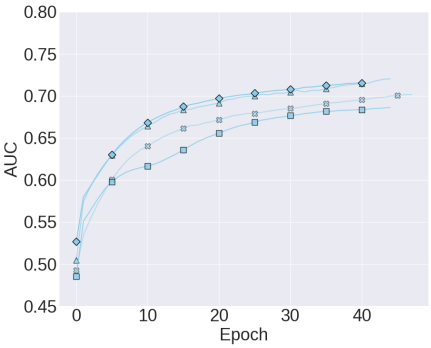


(g) BALC_{KLD}

(h) Temporal BALC_{KLD}



(i) BALC_{JSD}



(j) Temporal BALC_{JSD}

Figure 17: Mean AUC of the validation set as a function of acquisition epochs τ for the different acquisition functions using the MCP method. MC samples and the acquistion percentage were fixed at $T = 20$ and $b = 2\%$, respectively. These experiments are performed on $\mathcal{D}_2$ at a fraction $\beta = 0.5$. Results are averaged across 5 seeds.