Appendix

Baseline acquisition function details

Max entropy selects the points that maximize the predictive entropy

$$\alpha(x, \mathcal{M}) = H(y|x, \mathcal{D}_{train}) \\ = -\sum_c p(y = c|x, \mathcal{D}_{train}) \log(p(y = c|x, \mathcal{D}_{train}))$$

BatchBALD BatchBALD [Kirsch et al., 2019] tries to find a batch of points that has the highest mutual information with respect to the model parameters. **BALD** is the non-batched version of BatchBALD. Formally

$$\alpha_{BatchBALD}(\{x_1, \ldots, x_B\}, p(\omega)) \\ = H(y_1, \ldots, y_B) - \mathbb{E}_{p(\omega)}[H(y_1, \ldots, y_B|\omega)]$$

Filtered active submodular selection (FASS) FASS [Wei et al., 2015] samples the $\beta \times B$ most uncertain points $\mathcal{B}'$ and then subselect B points that are as representative of $\mathcal{B}'$ as possible. For the measure of uncertainty, FASS uses entropy $H(y|x, \mathcal{D}_{train})$. To measure the representativeness of $\mathcal{B}$ to $\mathcal{B}'$, FASS tries to choose $\mathcal{B}$ to maximize the following function

$$f(\mathcal{B}) = \sum_{y \in \mathcal{Y}} \sum_{i \in V^y} \max_{s \in \mathcal{B} \cap V^y} w(i, s)$$

Here $V^y \subseteq \mathcal{B}'$ is the set of points in $\mathcal{B}'$ with predicted label, y and $w(i, s) = d - ||x_i - x_s||_2^2$ is the similarity function between points indexed by i, s where $x_i, x_s \in \mathcal{X}$ and d is the maximum distance between two points. The idea here is that if a point in $\mathcal{B}$ already exists that is close to some point $x' \in \mathcal{B}'$, then $f(\mathcal{B})$ will favor adding points to the batch that are close to points other than x', thus increasing the batch diversity. Note that FASS is equivalent to Max Entropy if $\beta = 1$.

Bayesian Coresets In Pinsler et al. [2019], they try to build a batch such that the log posterior after acquiring that batch best approximates the complete data log posterior (i.e. the log posterior after acquiring the entire pool set). Their approach closely follows the general Bayesian Coreset [Campbell and Broderick, 2018] approach which constructs a weighted subset of data that approximates the full dataset. Crucially [Pinsler et al., 2019] assume that the posterior predictive distribution Y_p of a point p is independent of that of the corresponding distribution $Y_{p'}$ of another point p' – an assumption we do not make. We show in the next section why avoiding such an assumption lets us more effectively minimize the error with respect to the test distribution versus just optimizing for maxmizing information gain for the model posterior. As [Pinsler et al., 2019] require a *variable* batch size whereas all other methods (including ours) use a fixed batch size, for fairness of comparison, if the batch for this approach is smaller than the batch size being used, we fill the rest of the batch with random points. In practice, we only observe this being necessary for CIFAR.

Batch Active learning by Diverse Gradient Embeddings (BADGE) BADGE [Ash et al., 2019] tries to acquire points that are distant in hallucinated gradient space (for diversity) as well as have a high impact on the parameters of the final output layer (as a proxy for uncertainty).

Random The points are selected uniformly at random from the unlabeled pool. Thus $\alpha(x, \mathcal{M})$ is the uniform distribution.

Motivating example 2

Suppose we have a model distribution with 10 possible models $\omega_1, \ldots, \omega_{10}$ with equal prior probability of being the true model ($p(w_i) = 0.1 \quad \text{for } \forall i$). Let the datapoints be $x_1, \ldots, x_L$ with their labels taking 4 possible values. We define $p_{ij}^k = p(y_i = j | x_i, \omega_k)$ as the probability of the jth class for the ith datapoint given by the kth model. Let

$$
\begin{aligned}
&p_{1j}^k = 1; j = k, 1 \leqslant k \leqslant 3 \\
&p_{14}^k = 1; 4 \leqslant k \leqslant 10 \\
&p_{i1}^k = 1, p_{i2}^{10} = 1; 1 \leqslant k \leqslant 9, 2 \leqslant i \leqslant L
\end{aligned}
$$

	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
x_1	1	2	3	4	4	4	4	4	4	4
$x_2 \ldots x_L$	1	1	1	1	1	1	1	1	1	2

Table 2: Labels that the different points x_i take with probability 1 under different models. The columns are the different models ω_k, and the rows are the different points.

Given that we have no other information about the models, we update the posterior probabilities for the models as follows – if a model ω_k outputs label l for a point x but after acquisition, the label for x is not l, then we know that is not the correct model and thus its posterior probability is 0 (so it is eliminated). Otherwise we have no way of distinguishing between the remaining models so they all have equal posterior probability. Then for x_1 the mutual information is

$$
\begin{aligned}
&\mathbb{I}[y_1, \omega | x_1, \mathcal{D}_{train}] \\
&= H[y_1 | x_1] - \mathbb{E}_{p(\omega | \mathcal{D}_{train})}[H[y_1 | x_1, \omega]] = 0.94
\end{aligned}
$$

For $x_2 \ldots x_L$, $\mathbb{I}[y_{2-L}, \omega | x_{2 \ldots L}, \mathcal{D}_{train}] = 0.325$. However selecting x_1 would decrease the expected posterior entropy $H[y_{2-L} | x_{2 \ldots L}, x_1, y_1, \mathcal{D}_{train}]$ from 0.325 to only 0.287. Acquiring any of $x_{2 \ldots L}$ instead of x_1, however, would decrease that entropy to 0, which would cause a much larger decrease in the expected posterior entropy averaged over $x_{1 \ldots L}$ if L is large enough. The detailed calculations are in the later subsection.

While $x_{2 \ldots L}$ may not contribute much to the entropy of the *joint* predictive distribution or to the MI with respect to the model parameters compared to x_1, collectively they will be weighted $L - 1$ times more than x_1 when looking at the accuracy. We should thus expect a well-calibrated model to have a higher uncertainty, and thus make a lot more errors on $x_{2 \ldots L}$, if x_1 is acquired versus if any of $x_{2 \ldots L}$ are acquired. For instance, in the above example, as L increases, the expected error rate would approach $\approx 0.7 \times (1/7 \times 6/7) \times 2 = 0.17$ (0.7 as 0.3 of the times the value of x_1 would also fix what the true model is reducing error rate on all x to 0) if x_1 is acquired as the errors for $x_{2 \ldots L}$ are correlated, whereas the rate would approach 0 were any of $x_{2 \ldots L}$ to be acquired.

Derivation for Example 2

For x_1, the mutual information between the predicted label y_1 and model parameters is:

$$
\begin{aligned}
&\mathbb{I}[y_1, \omega | x_1, \mathcal{D}_{train}] \\
&= H[y_1 | x_1] - \mathbb{E}_{p(\omega | \mathcal{D}_{train})}[H[y_1 | x_1, \omega]] \\
&= H[\sum_{k=1}^{10} p(y_1 | x_1, \omega_k) p(\omega_k)] - \sum_{k=1}^{10} p(\omega_k) H[p(y_1 | x_1, \omega_k)] \\
&= -(3 \times (\frac{1}{10} \times \log(\frac{1}{10})) + \frac{7}{10} \times \log(\frac{7}{10})) \\
&- 10 \times \frac{1}{10} \times (-(1 \times \log(1) + 0 \times \log(0))) \\
&= 0.940
\end{aligned}
$$

For $x_{2...L}$,

$$\begin{aligned}
&\mathbb{I}[y_{2-L}, \omega | x_{2...L}, \mathcal{D}_{train}] \\
&= -(\frac{9}{10} \times \log(\frac{9}{10}) + \frac{1}{10} \times \log(\frac{1}{10})) \\
&- 10 \times \frac{1}{10}(-(1 \times \log(1) + 0 \times \log(0))) \\
&= 0.325
\end{aligned}$$

After acquiring x_1, assuming the true label for x_1 is 1, then we update the posterior over the model parameter such that $p'(w_1)|_{y_1=1} = 1$ and $p'(w_k)|_{y_1=1} = 0$ for $1 < k \leqslant 10$. Then the expected averaged posterior entropy for $x_{1...L}$ is:

$$\begin{aligned}
&\frac{1}{L-1} \sum_{i=2}^{L} H[y_i|x_i]|_{y_1=1} \\
&= \frac{1}{L-1} \sum_{i=2}^{L} H[\sum_{k=1}^{10} p(y_i|x_i, \omega_k) p'(\omega_k)|_{y_1=1}] \\
&= \frac{1}{L-1} \times (L-1) \times (-(1 \times \log(1) + 0 \times \log(0))) \\
&= 0
\end{aligned}$$

Similarly, we could compute the case where the true label for x_1 is 2-4:

$$\begin{aligned}
&\frac{1}{L-1} \sum_{i=2}^{L} H[y_i|x_i]|_{y_1=2} = 0 \\
&\frac{1}{L-1} \sum_{i=2}^{L} H[y_i|x_i]|_{y_1=3} = 0 \\
&\frac{1}{L-1} \sum_{i=2}^{L} H[y_i|x_i]|_{y_1=4} \\
&= \frac{1}{L-1} \times (L-1) \times (-(\frac{6}{7} \log(\frac{6}{7}) + \frac{1}{7} \log(\frac{1}{7}))) \\
&= 0.41
\end{aligned}$$

The expectation of the averaged posterior entropy with respect to predicted label for y_1 (since we don't know the true label) is:

$$\begin{aligned}
&H[y_{2-L}, \omega | x_{2...L}, x_1, y_1 \mathcal{D}_{train}] \\
&= \mathbb{E}_{y_1 \sim p(y_1|\mathcal{D}_{train})}[\frac{1}{L-1} \sum_{i=2}^{L} H[y_i|x_i]|_{y_1}] \\
&= \frac{1}{10} \times 0 + \frac{1}{10} \times 0 + \frac{1}{10} \times 0 + \frac{7}{10} \times 0.41 \\
&= 0.287
\end{aligned}$$

Further statistical background

A divergence Λ between two distributions is a measure of the discrepancy or difference between two distributions P, Q. A key property of a divergence is that it is 0 if and only if P, Q are the same distribution. In this paper, we will be using the KL divergence and the MMD, which are respectively defined as

$$\begin{aligned}
D_{KL}(P||Q) &= -\sum_{x \in \mathcal{X}} P(x) \log(\frac{Q(x)}{P(x)}) \\
MMD_k^2(P, Q) &= \mathbb{E}k(X, X') + k(Y, Y') - 2k(X, Y)
\end{aligned}$$

where k is a kernel in the Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ and μ_k is the mean embedding of the distribution into $\mathcal{H}$ as per the kernel k. We can then use the notion of divergence to define the *dependency* $\mathfrak{d}$ between a set of random variables $X_{1:n}$ as follows

$$\mathfrak{d}(X_{1:n}) = \Lambda(P_{1:n}, \otimes_i P_i)$$

where $P_{1:n}$ is the joint distribution of $X_{1:n}$, P_i the marginal of X_i with $\otimes P_i$ being the product of marginals. For D_{KL} the dependency is exactly MI as defined above. For MMD the dependency is the Hilbert-Schmidt Independence Criterion ($HSIC$).

Proof of Proposition 1

k^* is positive semidefinite (psd) and symmetric as the sum of psd symmetric matrices is also psd symmetric.

Proof of Proposition 2

We show here that

$$\widehat{dHSIC}(k^1, k^3, \ldots, k^d) + \widehat{dHSIC}(k^2, k^3, \ldots, k^d) = \widehat{dHSIC}(k^1 + k^2, k^3, \ldots, k^d)$$

but the extension to the arbitrary sums is straightforward. Here $\widehat{dHSIC}$ is the estimator for $dHSIC$ which is the d-variable version of $HSIC$. It is defined as

$$dHSIC = \frac{1}{n^2}\sum_{a=1}^{n}\sum_{b=1}^{n}\Pi_{j=1}^{d}k^j(X_{i_a}^j, X_{i_b}^j) + \frac{1}{n^{2d}}\Pi_{j=1}^{d}\sum_{a=1}^{n}\sum_{b=1}^{n}k^j(X_{i_a}^j, X_{i_b}^j) - \frac{2}{n^{d+1}}\sum_{a=1}^{n}\Pi_{j=1}^{d}\sum_{b=1}^{n}k^j(X_{i_a}^j, X_{i_b}^j)$$

where k^j is the kernel of the jth random variable and X_i^j is the ith observation for the jth random variable. The estimator $\widehat{dHSIC}$ is defined as [Sejdinovic et al., 2013a]

$$\widehat{dHSIC} = \frac{1}{n^2}\sum_{a=1}^{n}\sum_{b=1}^{n}\Pi_{j=1}^{d}k^j(x_{i_a}^j, x_{i_b}^j) + \frac{1}{n^{2d}}\Pi_{j=1}^{d}\sum_{a=1}^{n}\sum_{b=1}^{n}k^j(x_{i_a}^j, x_{i_b}^j) - \frac{2}{n^{d+1}}\sum_{a=1}^{n}\Pi_{j=1}^{d}\sum_{b=1}^{n}k^j(x_{i_a}^j, x_{i_b}^j)$$

As $dHSIC$ reduces to $HSIC$ when $d = 2$, the proof for $HSIC$ also follows. Using the definition of $\widehat{dHSIC}$ above,

$$\widehat{dHSIC}(k^1, k^3, \ldots, k^d) + \widehat{dHSIC}(k^2, k^3, \ldots, k^d) = \frac{1}{n^2}\sum_{a=1}^{n}\sum_{b=1}^{n}k^1(x_{i_a}^1, x_{i_b}^1)\prod_{j=3}^{d}k^j(x_{i_a}^j, x_{i_b}^j)$$

$$
\begin{aligned}
&+ \frac{1}{n^{2d}} \sum_{a=1}^{n} (\sum_{b=1}^{n} k^1(x^1_{i_a}, x^1_{i_b})) \prod_{j=3}^{d} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \\
&- \frac{2}{n^{d+1}} (\sum_{a=1}^{n} \sum_{b=1}^{n} k^1(x^1_{i_a}, x^1_{i_b})) \prod_{j=3}^{d} \sum_{a=1}^{n} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \\
&+ \frac{1}{n^2} \sum_{a=1}^{n} \sum_{b=1}^{n} k^2(x^2_{i_a}, x^2_{i_b}) \prod_{j=3}^{d} k^j(x^j_{i_a}, x^j_{i_b}) \\
&+ \frac{1}{n^{2d}} \sum_{a=1}^{n} (\sum_{b=1}^{n} k^2(x^2_{i_a}, x^2_{i_b})) \prod_{j=3}^{d} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \\
&- \frac{2}{n^{d+1}} (\sum_{a=1}^{n} \sum_{b=1}^{n} k^2(x^2_{i_a}, x^2_{i_b})) \prod_{j=3}^{d} \sum_{a=1}^{n} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \\
&= \Big[\frac{1}{n^2} \sum_{a=1}^{n} \sum_{b=1}^{n} k^1(x^1_{i_a}, x^1_{i_b}) \prod_{j=3}^{d} k^j(x^j_{i_a}, x^j_{i_b}) \\
&+ \frac{1}{n^2} \sum_{a=1}^{n} \sum_{b=1}^{n} k^2(x^2_{i_a}, x^2_{i_b}) \prod_{j=3}^{d} k^j(x^j_{i_a}, x^j_{i_b}) \Big] \\
&+ \Big[\frac{1}{n^{2d}} \sum_{a=1}^{n} (\sum_{b=1}^{n} k^1(x^1_{i_a}, x^1_{i_b})) \prod_{j=3}^{d} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \\
&+ \frac{1}{n^{2d}} \sum_{a=1}^{n} (\sum_{b=1}^{n} k^2(x^2_{i_a}, x^2_{i_b})) \prod_{j=3}^{d} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \Big] \\
&- \Big[\frac{2}{n^{d+1}} (\sum_{a=1}^{n} \sum_{b=1}^{n} k^1(x^1_{i_a}, x^1_{i_b})) \prod_{j=3}^{d} \sum_{a=1}^{n} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \\
&+ \frac{2}{n^{d+1}} (\sum_{a=1}^{n} \sum_{b=1}^{n} k^2(x^2_{i_a}, x^2_{i_b})) \prod_{j=3}^{d} \sum_{a=1}^{n} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \Big] \\
&= \frac{1}{n^2} \sum_{a=1}^{n} \sum_{b=1}^{n} (k^1(x^1_{i_a}, x^1_{i_b}) + k^2(x^2_{i_a}, x^2_{i_b})) \prod_{j=3}^{d} k^j(x^j_{i_a}, x^j_{i_b}) \\
&+ \frac{1}{n^{2d}} \sum_{a=1}^{n} \Big[\sum_{b=1}^{n} (k^1(x^1_{i_a}, x^1_{i_b}) \\
&+ k^2(x^2_{i_a}, x^2_{i_b})) \Big] \prod_{j=3}^{d} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \\
&- \frac{2}{n^{d+1}} \Big[\sum_{a=1}^{n} \sum_{b=1}^{n} (k^1(x^1_{i_a}, x^1_{i_b}) \\
&+ k^2(x^2_{i_a}, x^2_{i_b})) \Big] \prod_{j=3}^{d} \sum_{a=1}^{n} \sum_{b=1}^{n} k^j(x^j_{i_a}, x^j_{i_b}) \\
&= d\widehat{HSIC}(k^1 + k^2, k^3, \ldots, k^d)
\end{aligned}
$$

7.1 Further scaling to large batch sizes

To scale to large batch sizes, instead of adding points to the batch to be acquired one at a time, we can add points in minibatches of size L. While this comes at the cost of possible diversity in the

batch, we find that the tradeoff is acceptable for the datasets we experimented with. This gives a final computation cost of $O(\frac{|\mathcal{D}_U|m^2B\cdot C}{L})$ where C is the number of classes. By contrast the corresponding runtime for BatchBALD is $O(\mathcal{D}_U| \cdot B \cdot C \cdot m \cdot m')$ where m' is the number of sampled configurations of $y_{1:n-1}$. For all experiments with ICAL, we were able to use $L = 1$ without any scaling difficulties. For ICAL-pointwise, we used $L = \frac{B}{15}$ only for CIFAR-10 and CIFAR-100. As alluded to previously, ICAL-pointwise can accommodate much larger L compared to ICAL before its performance degrades, allowing for much greater scaling. We evaluate this aspect of ICAL-pointwise in the Appendix.

The final algorithm is given in Algorithm 1.

7.2 Algorithm

Algorithm 1 Information Condensing Active Learning (ICAL) $(\mathcal{M}, T, \mathcal{D}_{train}, \mathcal{D}_U, B, K, r, L)$

Train $\mathcal{M}$ on $\mathcal{D}_{train}$
repeat
  $\mathcal{B} = \{\}$
  while $|\mathcal{B}| < B$ **do**
    Y^U = the predictive distribution for $x \in \mathcal{D}_U$ according to $\mathcal{M}$
    R = Set of r randomly selected points from $\mathcal{D}_U$
    $x' = \text{argmax}_x\, \alpha_{ICAL}(\mathcal{B} \cup \{x\}, HSIC)$ with the optimizations as specified in Section 5.1 and 5.2
    $\mathcal{B} = \mathcal{B} \cup \{x'\}$
  end while
  $\mathcal{D}_{train} = \mathcal{D}_{train} \cup \mathcal{B}$
  Retrain $\mathcal{M}$ on $\mathcal{D}_{train}$
until T iterations reached
Return $\mathcal{M}$

ICAL-pointwise

To evaluate the marginal dependency increase if a candidate point x is added to batch $\mathcal{B}$, we sample a set R from the pool set $\mathcal{D}_U$ and compute the pairwise $dHSIC$ of both $\mathcal{B}$ and $\mathcal{B}' = \mathcal{B} \cup \{x\}$ with respect to each point in R. Let the resulting vectors (each of length $|R|$) with the $dHSIC$ scores be $\mathfrak{d}_{\mathcal{B}}$ and $\mathfrak{d}_{\mathcal{B}'}$. Then the marginal dependency increase statistic M_x for point p is $M_x = \frac{1}{|R|}\sum_i \max((\mathfrak{d}^i_{\mathcal{B}'}/\mathfrak{d}^i_{\mathcal{B}}), 1)$ where i is the ith element of the vector. When then modify the α_{ICAL} as follows - $\alpha'_{ICAL}(\mathcal{B} \cup \{x\}) = \alpha_{ICAL}(\mathcal{B} \cup \{x\}) \cdot (M_x - 1)$ and use the point with the highest value of α'_{ICAL} as the point to acquire. Note that as we want to get as accurate an estimate of M_x as possible, we ideally want to choose as large a set R as possible. In general, we also want to choose $|R|$ to be greater than the number of classes. This makes ICAL-pointwise more memory intensive compared to ICAL. We also tried another criterion for batch selection based on the minimal-redundancy-maximal-relevance Peng et al. [2005] but that had significantly worse performance compared to ICAL and ICAL-pointwise.

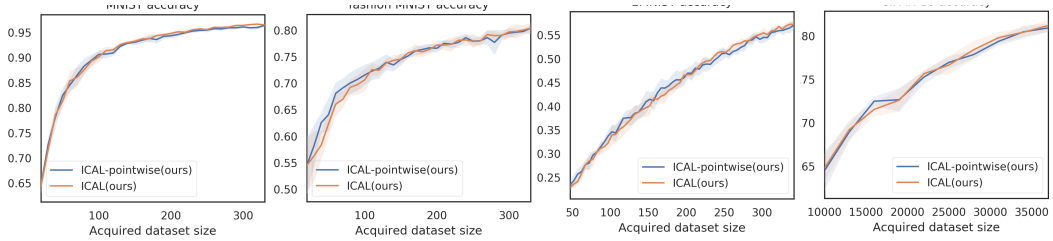

Figure 6: Relative performance of ICAL and ICAL-pointwise on smaller datasets (EMNIST,FashionMNIST,MNIST and CIFAR10) with parameters set to equivalent computation cost

In Figure 6, we analyze the performance of ICAL versus ICAL-pointwise when their parameters are set such that computational cost is about the same. As can be seen they are broadly similar with

ICAL-pointwise having a slight advantage in earlier acquisitions and ICAL being slightly better in later ones.

We also analyze the relative performance as the mini-batch size L changes in Figure 7. In the Figure, $iter = \frac{B}{L}$ is the number of iterations taken to build the entire acquisition batch (note that the actual acquisition happens *after* the entire batch has been built). ICAL-pointwise requires more computation time than ICAL in small L setup, however if time is the major constraint, ICAL-pointwise is to be preferred as its performance degrades more slowly as L, the size of the minibatch, increases. As the performance usually peaks at $L = 1$, if one is trying to get the best performance or if memory is a constraint, then ICAL is to be preferred.

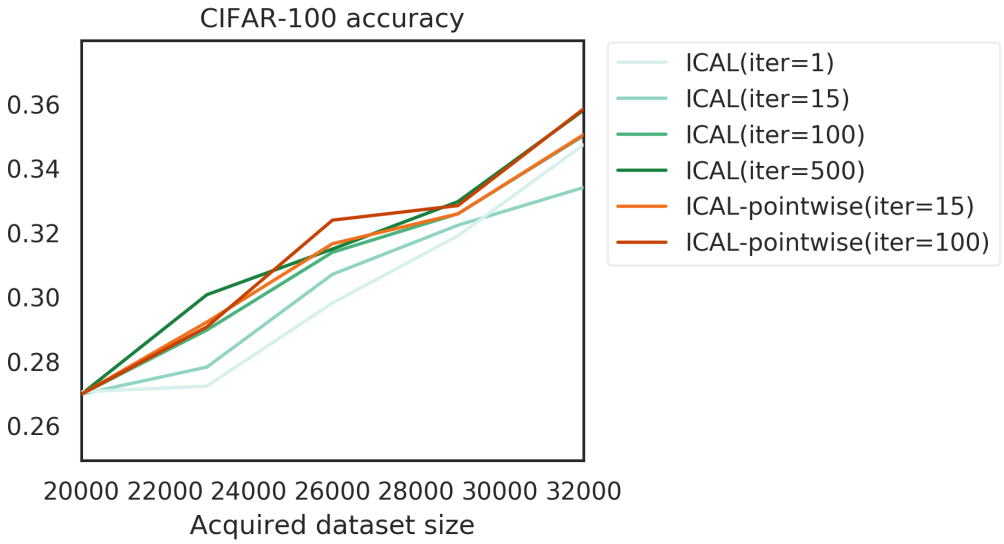

Figure 7: Relative performance of ICAL and ICAL-pointwise on CIFAR100 with different mini-batch size L. $iter = \frac{B}{L}$ is the number of iterations taken to build the entire acquisition batch of size B (note that the actual acquisition happens *after* the entire batch has been built)

Diversity of acquired samples in repeated-MNIST

To check if ICAL's acquisition batches are diversed enough, we plot the number of times different number of copies of a same sample has been acquired by each method. As shown in figure 8, our method (as well as BatchBALD, BayesCoreset and Random) successfully avoided acquiring redundant copies of the same sample, whereas FASS and Max Entropy acquired up to 3 copies of the same replica in most acquisitions. This proves that the batched active learning strategies are better in diversity.

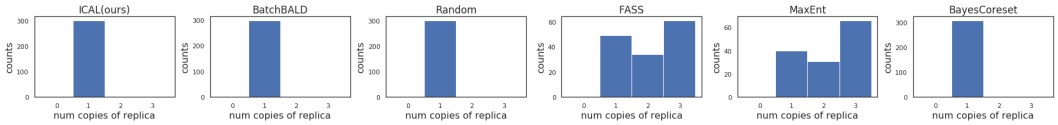

Figure 8: Frequencies where different numbers of copies (1-3) of a same sample has been acquired by each method.

Further CIFAR-10 and CIFAR-100 results

Further CIFAR results are in Table 1. For CIFAR-100, Random has a high p-value but that is mainly because it performs a bit better in the beginning vs. all other methods but its performance quickly degrades and it is far below ICAL in the final iteration.

Runtime and memory considerations

BatchBALD runs out of memory on CIFAR-10 and CIFAR-100 and thus we are unable to compare against it for those two datasets. For the MNIST-variant datasets, ICAL takes about a minute for building the batch to acquire (batch sizes of 5 and 10). For CIFAR-10 (batch size 3000), with $L = 1$, the runtime is about 20 minutes but it scales linearly with $1/L$ (Figure 10). Thus it is only 5 minutes for $L = 30$ ($iter = 100$) which is already sufficient to give comparable performance to $L = 1$ (Figure 9). For CIFAR-100 (batch size 3000), the performance does degrade with high L but as we mentioned previously, ICAL-pointwise holds up a lot better in terms of performance with high L (Figure 7) and thus if time is a strong consideration, that variant should be used instead.

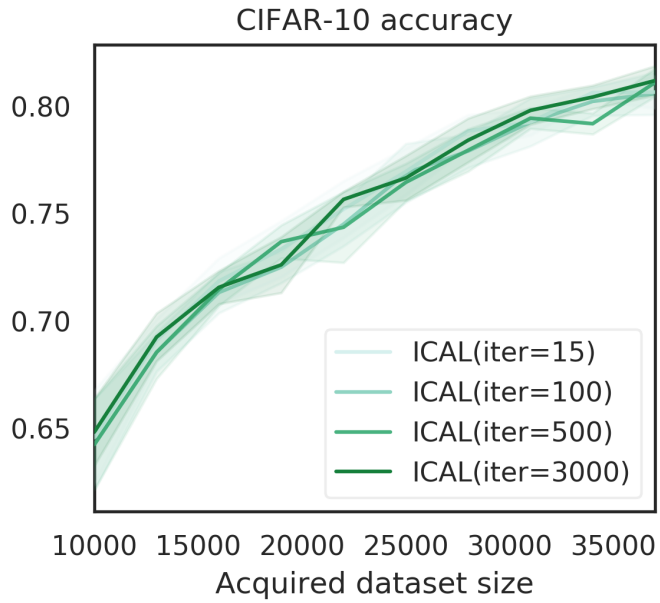

Figure 9: CIFAR10 performance with different L. $iter = \frac{B}{L}$ is the number of iterations taken to build the entire acquisition batch of size B (note that the actual acquisition happens *after* the entire batch has been built)

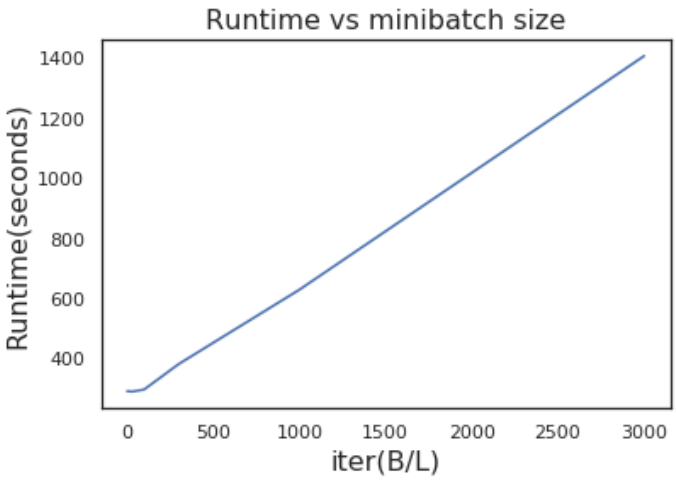

Figure 10: Runtime of ICAL on CIFAR10 with different minibatch size L.