A DERIVATIONS

Here, we show the equivalence of Eq. (3) and Eq. (4) for i.i.d. data:

$$\mathrm{h}[\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}}] - \mathrm{h}[\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}} \cup (x, y)] \tag{7}$$

Definition

$$= -\log \mathrm{p}(\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}}) + \log \mathrm{p}(\mathbf{y}^{\text{ho}} \mid y, \mathbf{x}^{\text{ho}}, x, \mathcal{D}_{\text{t}}) \tag{8}$$

Bayes rule

$$= -\log \mathrm{p}(\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}}) + \log \frac{\mathrm{p}(y \mid \mathbf{y}^{\text{ho}}, \mathbf{x}^{\text{ho}}, x, \mathcal{D}_{\text{t}})\, \mathrm{p}(\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, x, \mathcal{D}_{\text{t}})}{\mathrm{p}(y \mid \mathbf{x}^{\text{ho}}, x, \mathcal{D}_{\text{t}})} \tag{9}$$

Rearrange symbol order

$$= -\log \mathrm{p}(\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}}) + \log \frac{\mathrm{p}(y \mid x, \mathbf{y}^{\text{ho}}, \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}})\, \mathrm{p}(\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, x, \mathcal{D}_{\text{t}})}{\mathrm{p}(y \mid x, \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}})} \tag{10}$$

Independence assumptions

$$= -\log \mathrm{p}(\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}}) + \log \frac{\mathrm{p}(y \mid x, \mathbf{y}^{\text{ho}}, \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}})\, \mathrm{p}(\mathbf{y}^{\text{ho}} \mid \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}})}{\mathrm{p}(y \mid x, \mathcal{D}_{\text{t}})} \tag{11}$$

Rearrange and cancel out

$$= \log \mathrm{p}(y \mid x, \mathbf{y}^{\text{ho}}, \mathbf{x}^{\text{ho}}, \mathcal{D}_{\text{t}}) - \log \mathrm{p}(y \mid x, \mathcal{D}_{\text{t}}) \tag{12}$$

Definition

$$= \mathrm{h}[y \mid x, \mathcal{D}_{\text{t}}] - \mathrm{h}[y \mid x, \mathbf{x}^{\text{ho}}, \mathbf{y}^{\text{ho}}, \mathcal{D}_{\text{t}}]. \tag{13}$$

B RELATED WORK: EXPECTED PREDICTIVE INFORMATION GAIN

Here, we provide a brief summary of the expected predictive information gain (EPIG) $\mathrm{I}[\mathbf{Y}^{\text{eval}}; Y \mid \mathbf{X}^{\text{eval}}, x], \mathcal{D}_{\text{t}}$ introduced by Kirsch et al. (2021) as it is relevant to our own notion of Predictive Information Content. EPIG is proposed in the context of (Bayesian) active learning, where no access to labels is assumed for the selection of informative datapoints. To emphasize the difference to our labelled held-out set, we adopt the authors' terminology of an unlabelled *evaluation set* $\{x^{\text{eval}}\}_i$. In the paper, the authors discuss the case of selecting a full batch, whereas we limit ourselves to the case of a single datapoint (x, y) for better comparability, i.e. the case of batch size $n_b = 1$.

Similar to our case, the aim is to maximize the expected information gain for these evaluation points. Given that no label access is assumed, this is approximated by the expectation over the predictions of y^{eval} and y. Furthermore, the expectation over the evaluation samples can be approximated using the joint, i.e. by

$$\mathbb{E}_{\mathrm{p}_{\text{eval}}(x^{\text{eval}}, y^{\text{eval}})} \left[\mathrm{h}[y^{\text{eval}} \mid x^{\text{eval}}, \mathcal{D}_{\text{t}} \cup (x, y)] \right] \tag{14}$$

$$\approx \mathbb{E}_{\mathrm{p}_{\text{eval}}(x^{\text{eval}})} \left[\mathrm{H}[Y^{\text{eval}} \mid x^{\text{eval}}, Y, x, \mathcal{D}_{\text{t}}] \right] \tag{15}$$

$$\geq \frac{1}{|\mathcal{D}_{\text{eval}}|} \mathrm{H}[\mathbf{Y}^{\text{eval}} \mid \mathbf{x}^{\text{eval}}, Y, x, \mathcal{D}_{\text{t}}] \tag{16}$$

where $\mathrm{H}[\cdot \mid \cdot]$ denotes the conditional entropy. It is noted that the bound is not tight in general but only in the infinite data limit when predictions become uncorrelated. That is, the authors introduce an approximation of EPIG as the expected information gain between the predictions Y for the candidate samples and the predictions for the evaluation set,

$$\mathrm{I}[\mathbf{Y}^{\text{eval}}; Y \mid \mathbf{x}^{\text{eval}}, x, \mathcal{D}_{\text{t}}] = \mathrm{H}[\mathbf{Y}^{\text{eval}} \mid \mathbf{x}^{\text{eval}}, \mathcal{D}_{\text{t}}] - \mathrm{H}[\mathbf{Y}^{\text{eval}} \mid \mathbf{x}^{\text{eval}}, Y, x, \mathcal{D}_{\text{t}}] \tag{17}$$

Since the first term $\mathrm{H}[\mathbf{Y}^{\text{eval}} \mid \mathbf{x}^{\text{eval}}, \mathcal{D}_{\text{t}}]$ is constant for a fixed $\mathcal{D}_{\text{t}}$, maximizing the approximation is equivalent to minimising the second term. Similar to our approach, it is suggested to estimate these quantity by expanding it in the opposite direction, as

$$\mathrm{I}[\mathbf{Y}^{\text{eval}}; Y \mid \mathbf{x}^{\text{eval}}, x, \mathcal{D}_{\text{t}}] = \mathrm{H}[Y \mid x, \mathcal{D}_{\text{t}}] - \mathrm{H}[Y \mid x, \mathbf{Y}^{\text{eval}}, \mathbf{x}^{\text{eval}}, \mathcal{D}_{\text{t}}]. \tag{18}$$

Since it still contains an expectation over the predicted labels, this is much harder to evaluate that in our setting where access to labels is given. The authors then suggest a way to do so.

C EXPERIMENT DETAILS

For experiments on QMNIST, we use a multi-layer perceptron with 2 hidden layers and 512 units in each hidden layer. For experiments on CIFAR-10, CIFAR-100, CINIC-10 and Clothing-1M we use a ResNet-18 (He et al., 2016). All models are trained using the Adam optimizer with default PyTorch hyperparameters (β_1=0.9, β_2=0.999, and weight decay of 0.01, learning rate 0.001), a training batch size $n_b = 32$ (64 for CINIC-10) with $n_B = 320$ (640 for CINIC-10) points pre-sampled, meaning we select $\frac{n_b}{n_B} = 10\%$ of points. On all datasets except QMNIST, we train using data augmentation (random crop and horizontal flip). The irreducible loss models are trained on holdout sets (not test sets, see above). For each dataset, we select the irreducible loss model from the epoch with lowest held-out loss on $\mathcal{D}$ (as opposed to highest accuracy). The multiple target architectures in Fig 5 were adapted from Phan (2021).

D IRREDUCIBLE HELD-OUT LOSS APPROXIMATION

To empirically justify the approximation made in Eq. 5, we train an irreducible loss model with limited capacity to improve when updated on $\mathcal{D}_t$. We make use of MNIST-8M (Loosli et al., 2007) and use an MLP with 2 hidden layers and 16 hidden units. MNIST-8M applies pseudo-random deformations to MNIST to increase the dataset to over 8 million images. We use the original MNIST training set to train the irreducible loss model and MNIST-8M training set, excluding the original MNIST training set, as $\mathcal{D}_t$. Note that the holdout set is thus much smaller than $\mathcal{D}_t$.

The accuracy of the irreducible loss model, with constrained capacity, remains at around 94% despite being updated on $\mathcal{D}_t$, and the updating has no discernible effect on the accuracy of the model trained exclusively on $\mathcal{D}_t$.

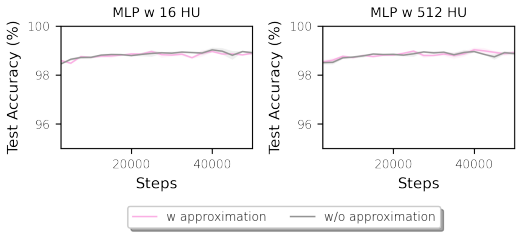

Figure 6: Reducible loss selection with and without the approximation in Eq. 5.

E SMALL IRREDUCIBLE LOSS MODELS AVOID NOISY POINTS

Here, we show one reason why selection with a small irreducible loss model still accelerates training: it still avoids points with noisy labels. Figure 7 is identical to Figure 1B, but includes selection using an irreducible loss model that uses 21x fewer parameters (and 29x fewer FLOPs) than the target model. It also has a simpler architecture (vanilla CNN versus ResNet-18). Across our datasets, selection with the smaller irreducible loss model avoids points whose labels are corrupted with uniform noise.

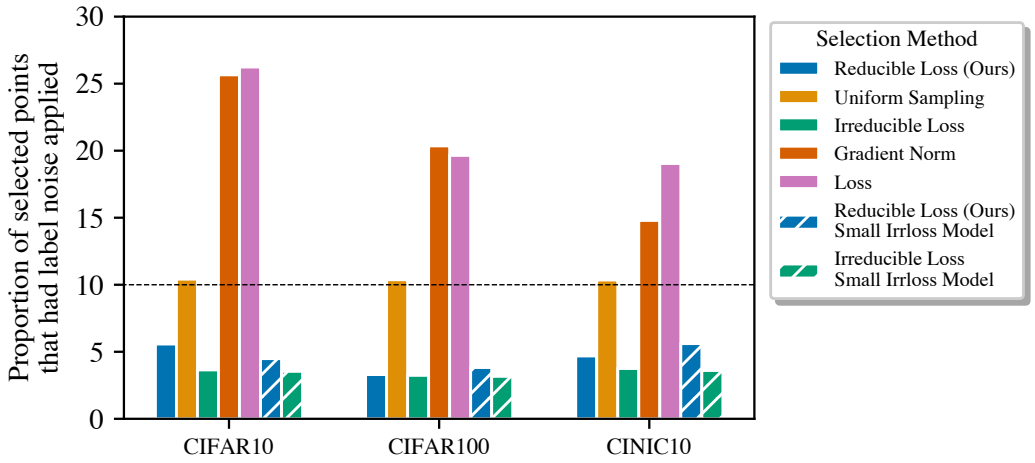

Figure 7: Proportion of selected points with label noise applied for different selection method. We added 10% uniform label noise, i.e., we randomly switched each point's label with 10% probability. Selecting the 10% "hardest" points (high loss or gradient norm) gives batches where $> 10\%$ of the labels are mislabeled. We can avoid noisy points by selecting points that are "easy" (low irreducible loss) or have high reducible loss. This works even when approximating the irreducible loss with a model that uses 29x fewer FLOPs and has 21x fewer parameters than the target model to be trained. Results are averaged over 15 epochs.

F Ablation of Percentage Selected

Our method has a hyperparameter, the percentage $\frac{n_b}{n_B}$ of evaluated points which are selected for training. In the experiments above, this parameter was set to 0.1. We have not tuned this parameter, as we aim to analyse how well our method works "out of the box". In fact, on some datasets, performance further improves with other values of this parameter settings of this hyperparameters seem to work better to give the highest speed. Adjusting this percentage should allow practitioners to specify their preferred tradeoff between training time and computation, where a low percentage corresponds to a lower training time. Training further accelerates when we lower the percentage from 10% to 5% and decelerates when we increase it to 20% or 50%. For these experiments, we keep $n_b = 32$ and adapt n_B accordingly. The percentage $\frac{n_b}{n_B}$ of datapoints selected per batch has different effects across datasets as shown in Figure 8.

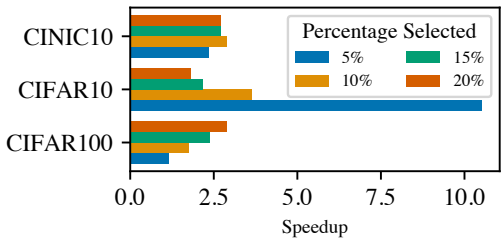

Figure 8: Varying the percent of data points selected in each training batch. Average over 3 random seeds.

G Active Learning Baselines

We compare our method to typical methods used in the Active Learning (AL) literature. Note that our method is label-aware, while active learning acquires datapoints without using labelled information. We consider the following baselines, which select the top-k points using an acquisiton function, $\alpha(x)$:

- Bayesian Active Learning by Disagreement (Houlsby et al., 2011) with $\alpha(x) = \mathrm{H}[y \mid x, \mathcal{D}_t] - \mathbb{E}_{\mathrm{p}(\theta|\mathcal{D}_t)}[\mathrm{H}[y \mid x, \theta]]$.
- (Average) conditional entropy, $\alpha(x) = \mathbb{E}_{\mathrm{p}(\theta|\mathcal{D}_t)}[\mathrm{H}[y \mid x, \theta]]$, where the average is taken over the model parameter posterior.
- (Average) entropy, $\alpha(x) = \mathrm{H}[y \mid x, \mathcal{D}_t]$.
- Loss minus conditional entropy $\alpha(x) = \mathrm{h}[y \mid x, \theta] - \mathbb{E}_{\mathrm{p}(\theta|\mathcal{D}_t)}[\mathrm{H}[y \mid x, \theta]]$. This uses the (average) conditional entropy as an estimate of how noisy datapoint x is—points with high noise are deprioritised.

We additionally compare our method to uniform sampling. We run all baselines on MNIST and CIFAR10. Note that several of these active learning baselines consider epistemic uncertainty; that is, uncertainty in predictions driven by uncertainty in the model parameters. This mandates performing (approximate) Bayesian inference. We use Monte-Carlo Dropout(Gal & Ghahramani, 2016) to perform approximate inference. For MNIST, we use an 2 hidden layer MLP with 512 hidden units per hidden layer, and a dropout probability of a 0.5. For experiments on CIFAR10, we use a small-scale CNN with dropout probability 0.05 (the dropout probability follows (Osawa et al., 2019)).

Figure 9 shows training curves for our method, uniform sampling, and the active learning baselines. Our method accelerates training across both datasets. The active learning methods accelerate training for MNIST but not for CIFAR10. This highlights that active learning methods, if naively applied to online batch selection, may not accelerate model training.

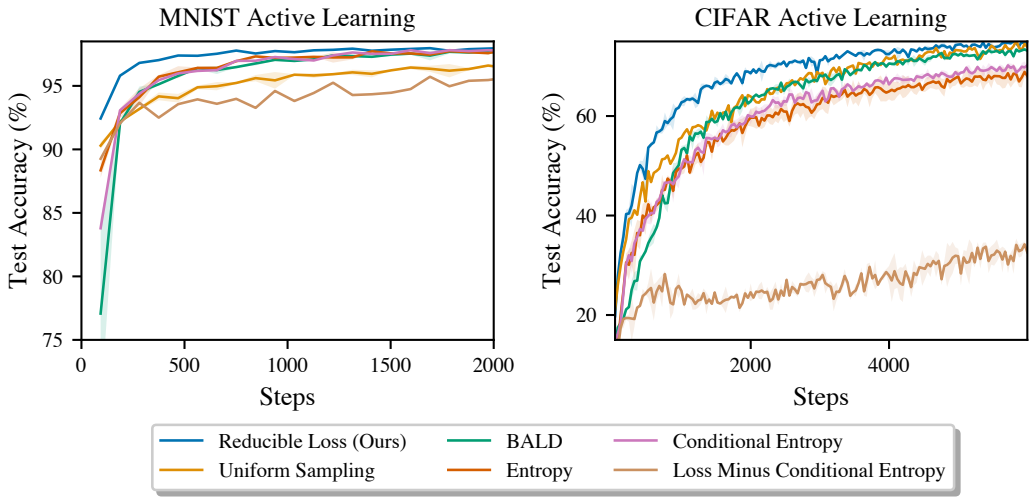

Figure 9: Training curves for several active learning baselines on the MNIST and CIFAR10 datasets.