VaB-AL: Incorporating Class Imbalance and Difficulty with Variational Bayes for Active Learning

Jongwon Choi*[1,2§], Kwang Moo Yi*[3], Jihoon Kim[1], Jincho Choo[1], Byoungjip Kim[1], Jin-Yeop Chang[1], Youngjune Gwon[1], and Hyung Jin Chang[4]

[1] Samsung SDS
[2] Graduate School of Advanced Imaging Science, Multimedia & Film, Chung-Ang University
[3] Department of Computer Science, University of Victoria
[4] School of Computer Science, University of Birmingham

Abstract. Active Learning for discriminative models has largely been studied with the focus on individual samples, with less emphasis on how classes are distributed or which classes are hard to deal with. In this work, we show that this is harmful. We propose a method based on the Bayes' rule, that can naturally incorporate class imbalance into the Active Learning framework. We derive that three terms should be considered together when estimating the probability of a classifier making a mistake for a given sample; i) probability of mislabelling a class, ii) likelihood of the data given a predicted class, and iii) the prior probability on the abundance of a predicted class. Implementing these terms requires a generative model and an intractable likelihood estimation. Therefore, we train a Variational Auto Encoder (VAE) for this purpose. To further tie the VAE with the classifier and facilitate VAE training, we use the classifiers' deep feature representations as input to the VAE. By considering all three probabilities, among them especially the data imbalance, we can substantially improve the potential of existing methods under limited data budget. We show that our method can be applied to classification tasks on multiple different datasets – including one that is a real-world dataset with heavy data imbalance – significantly outperforming the state of the art.

Keywords: active learning, Bayesian formulation, class imbalance, variational auto-encoder, deep learning

1 Introduction

Active learning focuses on efficient labelling of data and has drawn much interest lately [38, 41, 48], due to deep learning being attempted at new domains, such as biomedical imaging [3, 16] and industrial imaging [27, 49], where acquiring data can be costly [39]. Even for cases where data is not scarce, the effective usage of data may reduce training time, therefore the computational cost, including carbon foot-prints required to train each model. There have been various studies based on semi-supervised [7, 21] and unsupervised [37, 46] learning schemes to improve the training efficiency of data. However, with limited labelling budget, the performance of the studies are significantly

* These authors contributed equally to this work.
§ This paper was written when he worked for Samsung SDS.

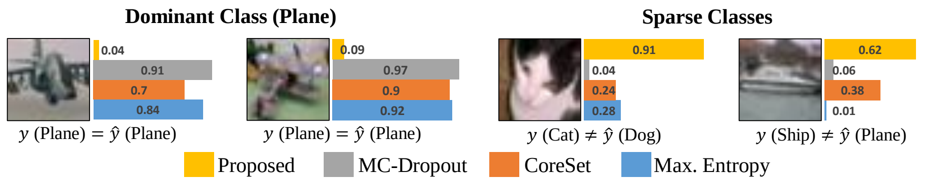


Fig. 1. Class matters – We show an example where existing methods fail to identify important samples to put into the updated training set. We show the importance metrics given by various methods – MC-Dropout [10], CoreSet [38], Max. Entropy [45], and our method – for selected images when the dataset is dominated by plane images. We further show below each image, the ground-truth class y and the predicted class $\widehat{y}$. Because of the data imbalance, existing methods heavily favour the plane class, even when the results are correct, as seen on the left. And for rare classes in the right column, they fail to recognise that the samples are important, which leads to the improved performance On the other hand, our method correctly identifies them as important, by considering also the class difficulty and the class prior.

worse to the supervised learning with the additionally labelled data [35]. In other words, their label efficiency could be improved.

Existing methods [4, 10, 38, 41, 48], regardless of how they are formulated, have a common underlying assumption that all classes are equal – they do not consider that some classes might just be harder to learn compared to others, or some classes might be more prevalent in the dataset than others. Instead, they focus on, given a data sample, how much error a trained model is expected to make, or the estimated uncertainties [10, 48]. These assumptions could be harmful, as in practice, since data is often imbalanced and not all classes are of the same difficulty [1, 51]. This can create a bias in the labelled data pool, leading to the trained classifier and active learning methods also being biased in deciding which samples to the label. As we show in Fig. 1, this can damage the capabilities of an active learning method significantly, even for typical benchmark datasets [6, 22].

In this work, we present a novel formulation for active learning, based on the classical Bayes' rule that allows us to incorporate multiple factors of a classification network together. Through derivation, we show that the probability of a classifier making mistakes can be decomposed into three terms; i) the probability of misclassification for a given predicted class, ii) the likelihood of a sample given predicted class, and iii) the prior probability of a class being predicted. In other words, one needs to take into account i) the difficulty of a class, ii) the performance of the classifier and iii) the abundance of a certain class of data holistically when determining the potential of a classification error. We take all of them into account and choose samples to be labelled by selecting those that have the highest misclassification probability.

While the task is discriminative, our method requires the estimation of likelihood, which could be intractable. We, therefore, propose to use a Variational Auto Encoder (VAE) [20] to model the lower bound of the likelihood of a sample. To make VAE conditioned on a predicted label, a naive way would be applied to train multiple VAEs for each predicted class. However, this quickly becomes impractical with a large number of classes. We thus propose to train a single VAE, with regularisation that acts as conditioning based on the predicted label. To further tie the VAE with the classifier, and for quick training of the VAE, we use the deep feature representations of the classifier as inputs to the VAE. Being generative, this training of VAE does not involve any labels, and we utilise the vast amount of unlabelled data with their predicted labels while inferring probabilities that are independent of data samples on the labelled ones.

We show empirically in Section 4 that our method allows a significant leap in performance for the benchmark dataset and the real application dataset, especially when the labelling budget is highly limited. Furthermore, we posit that considering the prior, that is, the distribution of labels during training, is critical when analysing the uncertainty of estimates.

In summary, our contributions are four-fold:

- we derive a novel formulation for active learning based on the Bayes' rule and posterior probability;
- we propose a framework based on VAE that realises this formulation;
- we reveal that considering the differences between classes – abundance and difficulty – is important;
- we outperform the state of the art in the various experiments.

2 Related Works

Traditional Active Learning. Increasing the label efficiency, thus reducing the cost associated with obtaining labels, has been of interest for decades. Even before deep learning became popular, various methods were suggested towards this goal [39]. Methods were proposed to estimate the uncertainty of the unlabelled samples through: the probability of prediction [24], the difference between the best prediction and the second one [25, 30, 36], or the entropy covering all the possible classes [17, 40]. For support vector machine classifiers the methods were suggested to utilise the distance from the decision boundary, for both the classification task [26, 43] and the detection task [44]. The algorithms clustering and finding representative samples were also suggested as another way [2, 12, 32].

Discrete optimisation algorithms have been proposed to consider the relationship between the sampling result and the model performance [8, 11, 47]. In a voting scheme-based algorithm [31], multiple different models are trained by the labelled pool, which determines the next queries according to their disagreement. Despite these efforts, the classical approaches are geared towards simple features and may hold limitations when applying to a large deep network with many nonlinear estimations.

Active Learning for deep networks. Active learning algorithms for deep networks can be categorised into uncertainty-based methods and representation-based methods. The uncertainty-based methods aim to select the uncertain samples from the unlabelled data pool and annotate them to increase the labelled pool [4, 10, 48]. Yoo and Kwon [48] proposed to use a small auxiliary "module" network predicting the training loss of the baseline network that is being trained with the active learning scheme. They then select the samples that are expected to give high losses. While their method is similar to ours in that an additional network is trained, as they require a ground-truth loss value while training, the auxiliary network can only be trained with labelled data, creating yet another network that the performance depends on how data is sampled. In contrast, we train our VAE with unlabelled data since we only rely on the predicted labels from the baseline network during training, and result in stable performance even with few labelled data. Gal *et al.* [10] proposed a method based on the estimation of sample-wise posterior probability through a Bayesian deep learning [9] framework. The method can be implemented simply by locating several dropout layers in a deep network, but this increases training time significantly until the convergence. Beluch *et al.* [4] suggest an active sampling method that estimates the disagreement of the prediction by using multiple deep networks. The downside of their method is that, as they use multiple networks, the memory and the computational requirement increases proportionally.

The representation-based methods target on finding representative samples within the high-dimensional space that deep networks learn [38, 41]. Sener and Savarese [38] proposed the Coreset algorithm that determines representative samples by using the feature maps of the intermediate layers of a deep network, rather than the last layer. However, the optimisation method in the Coreset algorithm does not scale well as the number of classes and the number of unlabelled samples grows. To improve the scalability, Sinha *et al.* [41] proposed to map the high dimensional feature maps into a lower dimension through adversarial training. Unfortunately, being based on adversarial training, the method requires a large amount of training data for the mapping to be reliable.

Beyond them, hybrid approaches combine the best of both uncertainty-based and representation-based methods [34, 50]. Some works focused on a specific task: for example person re-identification [29] and a human pose estimation [28].

While our work is most similar to uncertainty-based methods, it falls into neither uncertainty-based nor representation-based methods. Contrary to the previous uncertainty-based works, we take into account characteristics that are not restricted to a single sample – we consider the class difficulty and class imbalance. Also unlike the representation-based methods, we are not aiming to find samples that are representative, but a global trend of samples that are predicted to belong to a certain class.

Semi-supervised learning with VAEs. As we utilise VAEs [20], we also briefly review works related to VAEs that associate them with labelled data. Since VAEs model the likelihood of data, Lee *et al.* [23] used them to identify out-of-distribution samples for each class. We are loosely inspired by them, as we also use conditioned VAEs. However, unlike them, we estimate one portion of our conditional probabilities in estimating the label correctness. M2 VAE models [19] and Conditional VAEs [42] have been proposed to model conditional distributions. They directly add the condition as an additional latent dimension that is trained independently with the other latent dimensions for the reconstruction. In contrast, we apply conditioning implicitly during training to represent the class information and the feature distribution in the same latent dimensions. In our early attempts, we were not able to obtain successful modelling for our application with the former.

3 Method

We first formally describe the active learning problem and detail how we select new samples to be labelled with the estimated correctness of predictions. We then derive our method via Bayes' rule and describe how our derivation can be implemented in practice through a VAE. Afterwards, we provide a summary of how we tie every component together as an active learning algorithm and provide implementation details.

3.1 Problem formulation

Active learning can be formulated as a problem of increasing the pool of labelled data at every round by labelling subsets of the unlabelled pool. Formally, given a pool of data $\mathcal{P}$, we keep a pool of labelled data $\mathcal{P}_L$ and unlabelled data $\mathcal{P}_U$, such that $\mathcal{P} = \mathcal{P}_L \cup \mathcal{P}_U$ and $\varnothing = \mathcal{P}_L \cap \mathcal{P}_U$. Then, for each active learning round r, we select $\mathcal{P}_S^{(r)}$ that is a subset of $\mathcal{P}_U$ with N_r samples according to a criterion that defines the active learning method, which is moved from $\mathcal{P}_U$ to $\mathcal{P}_L$. Thus, $\mathcal{P}_U^{(r+1)} = \mathcal{P}_U^{(r)} - \mathcal{P}_S^{(r)}$ and $\mathcal{P}_L^{(r+1)} = \mathcal{P}_L^{(r)} + \mathcal{P}_S^{(r)}$.

The core of active learning is how $\mathcal{P}_S^{(r)}$ is selected, which in our case is based on the probability of a model providing a wrong answer for the given data.

Active learning based on the hardest examples. The underlying idea of our method is that when acquiring data with a limited labelling budget, one should acquire those that have the highest probability of making wrong predictions [10,25,29,48]. Formally, if we let y denote the real label and $\widehat{y}$ the label predicted by a model, we find samples $\mathbf{x}$ with large

$$p(y \neq \widehat{y}|\mathbf{x}). \tag{1}$$

Unfortunately, this is not a probability distribution that can be modelled directly, and we, therefore, estimate this probability via Bayes' rule and approximations.

3.2 Bayesian active learning

We now derive our Bayesian formulation. To estimate $p(y \neq \widehat{y}|\mathbf{x})$, we take an alternative route through Bayes' rule, instead of the direct estimation that could be given by a discriminative model. We first represent this probability with its complement, which can then be written as the sum of joint probabilities. We write

$$p(y \neq \widehat{y}|\mathbf{x}) = 1 - p(y = \widehat{y}|\mathbf{x}) = 1 - \sum_{n=1}^{N_c} p(y_n, \widehat{y}_n|\mathbf{x}), \tag{2}$$

where N_c is the number of classes and $p(y_n)$ and $p(\widehat{y}_n)$ is a shorthand for $p(y = n)$ and $p(\widehat{y} = n)$, respectively. Then, each $p(y_n, \widehat{y}_n|\mathbf{x})$ within the summation can be written through Bayes' rule as following:

$$p(y_n, \widehat{y}_n|\mathbf{x}) = p(y_n|\widehat{y}_n, \mathbf{x})p(\mathbf{x}|\widehat{y}_n)p(\widehat{y}_n)/p(\mathbf{x}). \tag{3}$$

Note that $p(\mathbf{x})$ is a uniform probability when each data point in the pool of datasets is unique. As we are interested in which sample gives maximum incorrectness, we can safely drop $p(\mathbf{x})$. Therefore, we have

$$p(y_n, \widehat{y}_n|\mathbf{x}) \propto p(y_n|\widehat{y}_n, \mathbf{x})p(\mathbf{x}|\widehat{y}_n)p(\widehat{y}_n). \tag{4}$$

Here, $p(y_n|\widehat{y}_n, \mathbf{x})$ corresponds to the probability of the real label being n, given that the predicted label is n and the data being $\mathbf{x}$. However, this is a probability that cannot be evaluated unless we have the paired true label. Thus, we instead do an informed guess, by ignoring $\mathbf{x}$ and approximating with $p(y_n|\widehat{y}_n)$. In other words, we assume that the probability of a model making a mistake is highly related to the label. Thus we approximate by writing

$$p(y_n, \widehat{y}_n|\mathbf{x}) \propto p(y_n|\widehat{y}_n)p(\mathbf{x}|\widehat{y}_n)p(\widehat{y}_n). \tag{5}$$

Finally, with Eq. (2), we have

$$p(y \neq \widehat{y}|\mathbf{x}) \propto -\sum_{n=1}^{N_c} p(y_n|\widehat{y}_n)p(\mathbf{x}|\widehat{y}_n)p(\widehat{y}_n). \tag{6}$$

Note that here, i) $p(y_n|\widehat{y}_n)$ is the probability of a model making mistake based on label, ii) $p(\mathbf{x}|\widehat{y}_n)$ is the *likelihood* of a sample given predicted label, iii) $p(\widehat{y}_n)$ is the *prior* on the distribution of predicted labels, which represents how imbalanced the predictions of a model are. As mentioned earlier in Section 1, the likelihood estimation is non-trivial and requires a generative model. We now detail how we model these three probabilities.

3.3 Estimating probabilities with regularized VAE

To estimate the probabilities we use a VAE [20]. Before we discuss the details, let us first clarify that we train this VAE exclusively in the unlabelled data pool $\mathcal{P}_U$, and associate it with the behaviour of the discriminative model on unseen data. We do not use the labelled pool $\mathcal{P}_L$, as it is likely that the classifier has overfitted to the data. Note also that while we explain in terms of $\mathbf{x}$, in fact, we use the deep feature representations given by the classifier to tie the VAE more with the classifier and to facilitate training by removing the need of learning the deep features. See Fig. 2 for an illustration of our framework.

We first detail why and how we estimate the likelihood with a VAE and then discuss the other two probabilities.

Likelihood of a sample – $p(\mathbf{x}|\widehat{y}_n)$. Estimating $p(\mathbf{x}|\widehat{y}_n)$ is not straightforward. A naive idea would be to implement multiple generative models that model $p(\mathbf{x})$, each trained with labels predicted to be of a certain class. However, this becomes quickly impractical as the number of classes grows. Moreover, estimating $p(\mathbf{x})$ can be intractable in practice [20].

In our work, we use a *single* VAE to estimate the lower bound of $p(\mathbf{x}|\widehat{y}_n)$ for all $\widehat{y}_n$. We use a VAE, as it learns to reconstruct the data sample using the tractable lower bound for $p(\mathbf{x})$. Distinct from existing work, to condition the $p(\mathbf{x})$ based on predicted labels, we propose to learn a latent space where the *absence* of parts of the latent embeddings

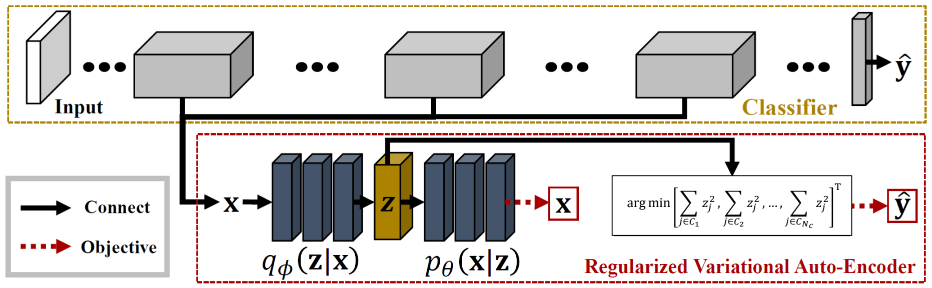

Input
Classifier
$\widehat{\mathbf{y}}$
Connect
Objective
$\mathbf{x}$
$\mathbf{z}$
$\mathbf{x}$
$q_\phi(\mathbf{z}|\mathbf{x})$
$p_\theta(\mathbf{x}|\mathbf{z})$
$\arg\min\left[\sum_{j\in C_1} z_j^2, \sum_{j\in C_2} z_j^2, \ldots, \sum_{j\in C_{N_c}} z_j^2\right]^\top$
$\widehat{\mathbf{y}}$
Regularized Variational Auto-Encoder

Fig. 2. Framework – An illustration of our framework. We train a Variational Auto Encoder (VAE) that models the deep feature representation of the classifier with unlabelled data. We then use the VAE to estimate $p(\mathbf{x}|\widehat{y}_n)$, $p(y_n|\widehat{y}_n)$, and $p(\widehat{y}_n)$, which in turn is used to estimate the probability of labelling error, $p(y \neq \widehat{y}|\mathbf{x})$. See Section 3.3 for details.

are related to the predicted label. In other words, this is as if we are training multiple VAEs with shared weights and overlapping latent spaces. Once such latent embeddings are learned, we compute the lower bound of $p(\mathbf{x}|\widehat{y}_n)$, by simply enforcing the absence manually via masking – thus selecting a VAE dedicated to a certain predicted class among the multiple virtual VAEs – and computing $p(\mathbf{x})$. This strategy allows us to have a manageable latent space, while still being able to deal with many classes.

In more detail, if we denote the j-th embedding dimension of VAE as z_j and write $j \in C_n$ to denote dimension j is related to class n, we write this absence condition as

$$\widehat{y} = \underset{n}{\operatorname{argmin}} \left[\sum_{j\in C_1} z_j^2, \sum_{j\in C_2} z_j^2, \ldots, \sum_{j\in C_{N_c}} z_j^2 \right]^\top . \tag{7}$$

Notice how this condition is conceptually similar to disentangled representations [5, 42]. In our earlier attempts, we have also tried forming this condition as disentangled representations, or enforcing $\sum_{j\in C_n} z_j^2$ to be zero if $\widehat{y}_n$, which neither was successful. We suspect that enforcing such constraints limit the capacity of the latent space and interferes with the training of the VAE too much. We have also tried other ways of enforcing absence – using the $\ell - 1$ norm or the sigmoid – but using the square worked best. We provide empirical results in Section 4.4.

We enforce this constraint as a form of regularisation. Let $\mathbf{w} = [w_1, w_2, \ldots, w_n]^\top$, where $w_n = \sum_{j\in C_n} z_j^2$, then we form an additional regularisation loss $\mathcal{L}_{\text{Class}}$ to be used during training of VAE as

$$\mathcal{L}_{\text{Class}} = \mathcal{H}\left(\text{softmax}\left(-\mathbf{w}\right), \widehat{\mathbf{y}}\right), \tag{8}$$

where $\mathcal{H}$ denotes the cross entropy, softmax is the softmax, and $\widehat{\mathbf{y}}$ is the one-hot encoded vector representation of $\widehat{y}$. With this regularisation term, recall from [20], that the training loss for VAEs $\mathcal{L}_{\text{VAE}}$ is the inverse of the empirical lower bound (ELBO) of the likelihood, which is defined as

$$\mathcal{L}_{\text{VAE}} = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] + D_{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}\right) \| \, p(\mathbf{z})\right), \tag{9}$$

where $p_\theta(\mathbf{x}|\mathbf{z})$ is the decoder with parameters θ and $q_\phi(\mathbf{z}|\mathbf{x})$ is the encoder with parameters ϕ. Therefore, the total loss to train our VAE is

$$\mathcal{L} = \mathcal{L}_{\text{VAE}} + \lambda\mathcal{L}_{\text{Class}}, \tag{10}$$

where λ is a hyperparameter that controls the regularisation strength. Note that with this loss, the VAE now also tries to mimic the behaviour of the classifier.

Once the VAE is trained, this VAE – without any conditioning – is now able to estimate the lower bound of the likelihood of a given data $p(\mathbf{x})$ [20] by simply computing the inversed value of $\mathcal{L}_{VAE}$. Furthermore, by masking the embedding space associated to $\widehat{y}$ with zero, we are able to compute the lower bound of $p(\mathbf{x}|\widehat{y})$. To avoid the decoder going too far from the latent embeddings it was trained for, we use Eq. (7) to obtain $\widehat{y}$, instead of the one from the classifier.

Probability of labelling error – $p(y_n|\widehat{y}_n)$. While the labelled pool $\mathcal{P}_L$ is the only set of data that we have labels for, as the trained classifier is likely to have overfitted to $\mathcal{P}_L$, we cannot use it for modelling this probability. We, therefore, use the labels given by the VAE for the data samples in the labelled pool $\mathcal{P}_L$. Note that the VAE has never seen these data points during training. Mathematically, if we denote the i-th sample as $\mathbf{x}^{(i)}$, its label as $y^{(i)}$, the predicted label from Eq. (7) as $\widehat{y}^{(i)}$, and introduce an indicator function δ that is 1 if all inputs are equal and 0 otherwise, we write

$$p(y_n|\widehat{y}_n) \approx \frac{\mathbb{E}_{\mathbf{x}\in\mathcal{P}_L, \mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\delta\left(y^{(i)}, \widehat{y}^{(i)}, n\right)\right]}{\mathbb{E}_{\mathbf{x}\in\mathcal{P}_L, \mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\delta\left(\widehat{y}^{(i)}, n\right)\right]}. \tag{11}$$

Here, we approximate the expectations with Monte Carlo estimates. Note that we also take the expectation over $\mathbf{z}$, to take into account the stochastic nature of VAEs.

Prior – $p(\widehat{y}_n)$. The prior is also acquired by using the labelled samples included in $\mathcal{P}_L$ as this probability should be related to how the classifier is trained. Same as in the case of $p(y_n|\widehat{y}_n)$, we cannot use the classifier predictions for the labelled pool, and we use the predictions from the VAE. Thus, sharing the notations as in Eq. (11), we write

$$p(\widehat{y}_n) \approx \mathbb{E}_{\mathbf{x}\in\mathcal{P}_L, \mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\delta\left(\widehat{y}^{(i)}, n\right)\right]. \tag{12}$$

3.4 Summary and implementation details

We summarise our method in Algorithm 1. For each Active Learning round r, we train a classifier $\mathcal{M}^{(r)}$ with the labelled pool $\mathcal{P}_L^{(r)}$. We then freeze the weights of the classifier, and train our VAE – $p_\theta^{(r)}(\mathbf{x}|\mathbf{z})$ and $q_\phi^{(r)}(\mathbf{z}|\mathbf{x})$ – with the unlabelled pool $\mathcal{P}_U^{(r)}$. We then estimate the three probabilities, which we use to construct the final estimate of $p(y \neq \widehat{y}|\mathbf{x})$, and sample those from $\mathcal{P}_U^{(r)}$ that has the highest value.

As noted earlier, we use the deep features of the baseline network (the classifier) as input to the VAE. Specifically, we extract deep features from the specific four layers of baseline. If the feature representations are from fully-connected layers, we use it as is. If it is from a convolutional layer, thus a feature map, we first apply global average pooling. We then apply batch normalisation to each feature, so that they are roughly in the same range, and feed it to a fully connected layer with 128 neurons and Sigmoid activation. Finally, the outputs from these fully-connected layers are concatenated to form a 512×1 vector and given as input to the VAE.

For the architecture of VAEs, we opt for a simple one as the deep features already have abstracted context. We apply four layers of fully connected layers with again 128 neurons and ReLU activations, with the exception of the layer that outputs the latent embeddings. For this layer, we use $10 \times N_c$ number of neurons and dedicate 10 dimensions per each class.

Algorithm 1: Proposed Method

```
Input: $\mathcal{P}_U^{(0)}$, $\mathcal{P}_L^{(0)}$, $N_r$, $\mathcal{M}$
$r$ = 0 while not at the maximum round do
    Train $\mathcal{M}$ using $\mathcal{P}_L^{(r)}$
    Freeze $\mathcal{M}$
    Train the VAE module using $\mathcal{P}_U^{(0)}$ by Eq. (10)
    Estimate $p(\mathbf{x}|\widehat{y}_n)$ by Eq. (9)
    Estimate $p(y_n|\widehat{y}_n)$ by Eq. (11)
    Estimate $p(\widehat{y}_n)$ by Eq. (12)
    Estimate $p(y \neq \widehat{y}|\mathbf{x})$ by Eq. (2)
    $\mathcal{X} \leftarrow N_r$ samples with the highest uncertainty
    $\mathcal{P}_U^{(r+1)} \leftarrow \mathcal{P}_U^{(r)} - \mathcal{X}$
    $\mathcal{P}_L^{(r+1)} \leftarrow \mathcal{P}_L^{(r)} \bigcup \mathcal{X}$
    $r \leftarrow r + 1$
end
```

We implement our method with PyTorch [33].To train our VAE, we set λ=0.005 in Eq. (10), for all tasks. We use Adam optimiser [18] with a learning rate of 10^{-4} and default parameters. We train for 20 epochs. To remove randomness from experiments, we perform our training multiple times with different initial conditions. We report both the average and the standard deviation of our experiments. For inference, to perform the Monte Carlo estimation in Eqns. (9), (11), and (12), we iterate the probabilistic inference 100 times for every sample.

4 Experimental Results

We first introduce the dataset, then the experimental setup including the classification network and the compared baselines. We then report our results and discuss the effect of the individual probability terms through an ablation study.

4.1 Dataset

To validate our experiments we apply our method to the standard *CIFAR-10* and *CIFAR-100* [22] datasets as well as the Northeastern University surface defect classification dataset (NEU) [14]. We use the CIFAR datasets to be comparable with recent Active Learning studies [41], and NEU to demonstrate that class imbalance is important in real-world applications. In more detail, the *CIFAR-10* dataset contains 60,000 images sized by 32 $\times$ 32 $\times$ 3 with 10 classes. The *CIFAR-100* dataset also consists of 60,000 images of the 32 $\times$ 32 $\times$ 3 size with 100 classes. Both the *CIFAR-10* and the *CIFAR-100* datasets are well balanced – all included images are assigned exactly one class among the 10 or the 100, and the number of instances per each class is equal. The datasets come with a pre-determined training and evaluation split of 50,000 and 10,000 images, respectively. The goal of the NEU dataset is to classify the 9 defect types on steels. The main feature of this dataset is that it is heavily imbalanced – the number of instances per class varies from 200 to 1589. In total, the NEU dataset contains 7226 defect images of size 64×64 pixel. This dataset does not provide pre-determined splits, and we thus randomly reserve 20% of the dataset for evaluation. This results in 5778 training samples and 1448 validation samples.

Introducing synthetic class imbalance. To demonstrate the importance of considering class-wise probabilities, we build additional variants of *CIFAR* datasets by removing a part of the samples in *CIFAR-10* and *CIFAR-100*. First, we build four datasets that have dominant classes within them – we hereafter refer to them as *dominant* datasets. A real-world example would be when scraping data from the internet – there will be many images of people, cats, and dogs, but not so many of platypus. The first three dominant datasets are built from *CIFAR-10* by randomly removing 90% samples of every category except for the $\{1, 5, 10\}$-th classes, respectively. For *CIFAR-100*, as there are many classes, there are only a few instances each – 500 for each class in the training set. We, therefore, build the last dominant dataset by removing 40% of the original samples for all categories other than the middle ones from 45-th class to 55-th class, from the *CIFAR-100* dataset. We denote these dominant datasets as *CIFAR-10*$^{+[1]}$, *CIFAR-10*$^{+[5]}$, *CIFAR-10*$^{+[10]}$, and *CIFAR-100*$^{+[45:55]}$, respectively.

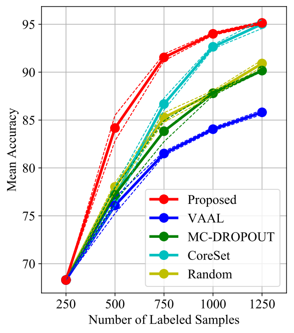
We further consider the case when some samples are rare. This would be, for example, cases where there are rare events, such as accidents or defects that need to be discovered. Similar to the *dominant* datasets, we build *rare* datasets by taking certain classes away from *CIFAR* datasets. Specifically, we use three variants, where we remove 90% of the samples that correspond to the $\{1, 5, 10\}$-th classes. We denote these as *CIFAR-10*$^{-[1]}$, *CIFAR-10*$^{-[5]}$, and *CIFAR-10*$^{-[10]}$, respectively.

With these datasets, we run different active learning setups, based on the size of the dataset. For NEU, we run five active learning rounds, where each round samples 250 samples. For CIFAR-10 based ones, we run six active learning round, which 500 samples each. For CIFAR-100, we run five rounds with 1000 samples each.

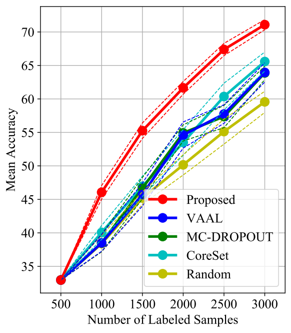
4.2 Experimental setup

The baseline classification network. As the baseline classifier we utilise ResNet-18 [13]. This network is composed of an initial simple convolution layer, followed by four basic residual blocks. As discussed previously in Section 3.4, the output feature maps from these four residual blocks are used to form the input to our VAE. At every Active Learning round, we train the network for 200 epochs, with a batch size of 128. We train with a typical setup for classification on CIFAR-10: SGD with the momentum of 0.9, weight decay of 0.0005, the learning rate is also initialised as 0.1 and decreased to 0.01 after 160 epochs. We repeat the same experiments with the different random seeds three times to remove the randomness from our experiments.

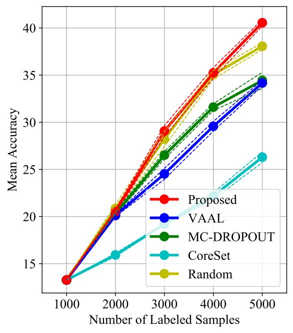
The competitors. To demonstrate the effectiveness of the proposed algorithm, we compare our method with the state-of-the-art for active learning. We consider *VAAL* [41], *MC-DROPOUT* [10], and *Core-set* [38]. We utilise the author's implementation for *VAAL* and *Core-set*. For *MC-DROPOUT* we integrate the author's implementation to our framework to utilise our baseline classification network – this method is architecture dependant. In more detail, we utilise the authors' code for uncertainty estimation and embed it into ours. Among the multiple methods to estimate the uncertainty in *MC-DROPOUT*, we use the Bayesian Active Learning by Disagreement (BALD) [15] as the



(a) NEU dataset



(b) *dominant* CIFAR-10



(c) *dominant* CIFAR-100

Fig. 3. Results for the dominant datasets.

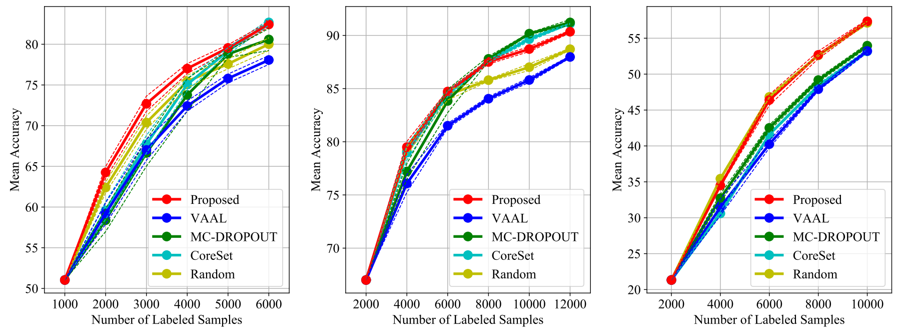
estimation method – it showed the best generality in our experiments. For the dropout layers of *MC-DROPOUT*, we place them before each the residual blocks of the ResNet and fix the dropout ratio to 0.25. As suggested by the authors [10], we further use 100 samples to estimate the uncertainty. In addition, we also consider uniform random sampling as a baseline. Finally, for a fair comparison, after new samples are extracted by these methods, we train the networks with the same random seed to further exclude any random surprises.

4.3 Comparison results

NEU dataset – a real-world imbalanced dataset. We first compare different methods on the NEU dataset, which is a typical case of an imbalanced real-world dataset. We report the results in Fig. 3(a). As shown, the proposed method delivers improved performance, especially for the early iterations. Methods here mostly converge after 5 rounds, as the dataset is relatively small. However, judging by the gap in early iterations, it could be expected that a similar gap would exist in a larger dataset. Interestingly, for this small dataset, VAAL performs worse than simple random selection. This is because the method requests a lot of labels to *bake in*.

***Dominant* datasets.** We now consider the dominant variants. In the dominant situation, a large part of the entire training samples is dedicated to a few specific categories – in the case of *CIFAR-10*, just one. In Fig. 3(b), we average the results from *CIFAR-10*$^{+[1]}$, *CIFAR-10*$^{+[5]}$, and *CIFAR-10*$^{+[10]}$ that we run three times each – a total of nine experiments. As shown, the proposed method outperforms the compared methods by a significant margin. This gap comes from the fact that the dataset is biased, and this causes the active learning methods to also become biased. However, our method successfully mitigates this class imbalance and provides improved performance.

A similar phenomenon happens when there are more classes. In Fig. 3(c), we report results for *CIFAR-100*$^{+[45:55]}$. Here, it is worth noting that all compared methods perform worse than *Random*. This is because the dataset has 100 classes, and it is easy to start



(a) *Rare* CIFAR-10 (b) *Full* CIFAR-10 (c) *Full* CIFAR-100

Fig. 4. Results for the rare and full datasets.

ignoring a certain class. Nonetheless, our method is the only method that provides improved label efficiency. To mitigate this, existing works have only studied when there is a sufficient number of labels from the beginning – for example in *VAAL* [41], 10% of the entire set. However, this greatly limits the applicability of active learning methods, as 10% is sometimes already too much to afford.

***Rare* datasets.** We further test the case when some classes are rare. In Fig. 4(a), we show the average performance of each method on the three rare datasets – *CIFAR-10*$^{-[1]}$, *CIFAR-10*$^{-[5]}$, and *CIFAR-10*$^{-[10]}$. Similar to the dominant case, our method shows the best performance among the compared methods. Interestingly, in this case, similar to the results with *CIFAR-100*$^{+[45:55]}$, existing methods perform worse than the random baseline. Again, this is because existing methods can break down easily without enough labelled samples from the beginning. This is a limitation our method does not have.

On the full dataset. For completeness, we further study how methods perform with the full *CIFAR-10* and *CIFAR-100* datasets. However note that, as we demonstrated previously, these datasets are with perfect data balance, which is not the case for real-world datasets. We report a summary of these results in Figures Fig. 4(b) and (c). Our method performs comparable to existing methods for *CIFAR-10* and outperforms existing methods for *CIFAR-100*. However, experiment for *CIFAR-100* shows a limitation of active learning methods including ours, that when there are too many classes and very few examples, their performances are close to random, and sometimes even worse. Nonetheless, compared to existing methods, our method performs best, even in this extreme situation.

Additional results. In the supplementary document, we further provide the original plots for all experiments before averaging. From them, we can confirm that the proposed method outperforms existing methods consistently for various situations.

Table 1. Results with and without the prior term $p(\widehat{y})$ and label difficulty term $p(y|\widehat{y})$.

$p(\widehat{y})$	$p(y\|\widehat{y})$	*CIFAR-10* avg.	*CIFAR-10* final	*CIFAR-10*$^{+[1]}$ avg.	*CIFAR-10*$^{+[1]}$ final
-	-	82.06%	88.86%	48.22%	60.56%
✓	-	82.92%	90.15%	54.54%	70.19%
-	✓	82.68%	90.02%	51.92%	66.25%
✓	✓	**82.96%**	**90.35%**	**54.77%**	**70.71%**

Table 2. Results with different ways of enforcing constraint on the latent space.

w_nType	*CIFAR-10*$^{+[1]}$ avg.	*CIFAR-10*$^{+[1]}$ final	*CIFAR-10*$^{+[5]}$ avg.	*CIFAR-10*$^{+[5]}$ final	*CIFAR-10*$^{+[10]}$ avg.	*CIFAR-10*$^{+[10]}$ final
$Sigmoid(z_j)$	41.78%	50.73%	40.49%	48.54%	41.69%	49.57%
$\|z_j\|$	45.00%	57.96%	44.40%	55.41%	44.31%	55.28%
z_j^2	**54.77%**	**70.71%**	**55.66%**	**72.02%**	**56.77%**	**70.53%**

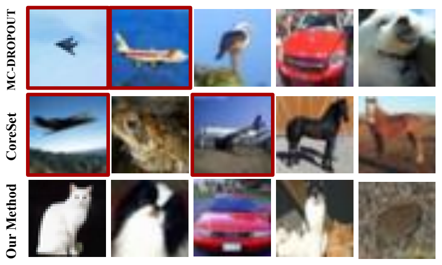


Fig. 5. Top samples for *CIFAR-10*$^{+[1]}$ – MC-DROPOUT (Top), CoreSet (second row), and our method (Third row). The dominant class *plane* is shown in red. Our method creates a balanced set by considering the class imbalance, whereas other methods are susceptible to it.

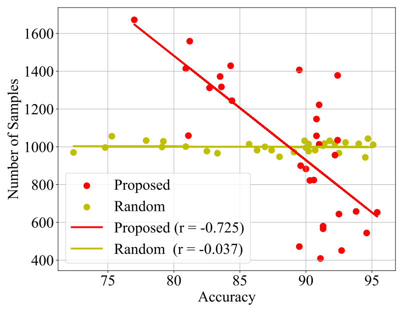
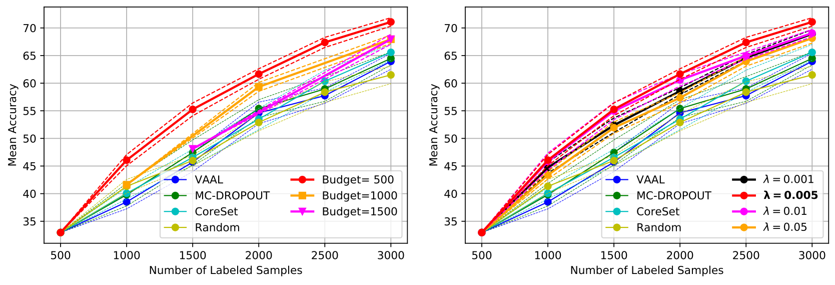


Fig. 6. Correlation between the class-wise accuracy and the class-wise sample sizes – We show a scatter plot of the accuracy of different classes vs the number of sample extracted for each class. For classes with high accuracy, our method samples less of them, focusing on the hard ones.

4.4 Ablation study

Effectiveness of $p(\widehat{y})$ and $p(y_n|\widehat{y})$. To validate their effectiveness, we perform an ablation study by excluding one or both of the prior ($p(\widehat{y})$) and the label difficulty ($p(y|\widehat{y})$) – we artificially set either to 1. We use *CIFAR-10* and *CIFAR-10*$^{+[1]}$ for these experiments to remove randomness. To avoid tuning on the test set, we use the validation splits for these experiments. We take the average performance over all active learning rounds and also the average final performance. We summarise the results in Table 1. We report both the average performance over all six active learning rounds (avg.) and the performance of the final round (final). We run each experiment three times and report the average. As shown, all terms contribute to performance improvement. We observe that all terms contribute to providing the best performance. Among the two terms, the prior ($p(\widehat{y})$) provides more gain compared to the label difficulty ($p(y|\widehat{y})$), demonstrating that it is critical to take data imbalance into account.

We show an example of the two terms in action in Fig. 5. Our method balances the training data even when the dataset is imbalanced. Besides the data imbalance, our method further considers the difficulty of each class, as demonstrated in Fig. 6.



(a) Results with varying budget sizes. (b) Results with varying λ.

Fig. 7. Results of ablation tests on *dominant* CIFAR-10.

Other choices for w_n To motivate our design decision, we further build two variants of our method, where we replace w_n in Eq. (7), either by a Sigmoid function ($Sigmoid(z_j)$) or a $\ell1$-norm ($|z_j|$). We summarise the results in Table 2. We observe that $\ell2$-norm outperforms the other two regularisation types for all compared cases. This is unsurprising, considering that in the case of $Sigmoid(z_j)$, it forces the latent embedding z_j to have extreme values, thus conflicting too much with the Gaussian distribution assumption that VAE aims to satisfy. This leads to the worst performance among the three types that we tried. In case of $|z_j|$, it would not suffer from this problem but would create constant gradients that are irrelevant to the magnitude of z_j, thus making it hard to enforce absence. Our choice, $\ell2$-norm, on the other hand, does not suffer from these shortcomings, becoming a natural choice for enforcing absence.

Robustness for various budgets and λ values To analyse the sensitivity of the proposed framework against the user-defined parameters, we report the result of our method with various sampling budgets and λ values in Eq. (10): see Fig. 7(a). Regardless of the choice of the budget size, our method outperforms all compared methods. Smaller budget size tends to allow methods to react faster to the training outcomes and increase the effectiveness of active learning.

In Fig. 7(b) we report the performance of our method with varying λ. We test with various λ, which do affect the performance of the proposed method to some degree. However, as before in the case of the budget, our method outperforms all compared methods regardless of the choice of λ. This further hints that the two-loss terms that we train for in Eq. (10) do not compete. Note that $\lambda = 0$ completely disables $p(\widehat{y})$ and $p(y|\widehat{y})$, as $\widehat{y}$ estimation becomes meaningless, which causes the method to perform significantly worse as shown earlier in Table 1, and we have therefore left it out in this figure to reduce cluster; see supplementary appendix for full results.

5 Conclusion

We have proposed a novel active learning method that incorporates class imbalance and label difficulty. Our method was derived through the Bayes' rule, which results in three types of probabilities – data likelihood, label prior, label difficulty – being considered together. We implement our method via a VAE, that is regularised to behave as a conditional VAE. We have shown that this creates a significant difference for a real-world dataset that exhibits data imbalance, as well as in cases when data imbalance is introduced to CIFAR-10 and CIFAR-100 datasets.

While we limit our experiments to classification in this work, our method is application agnostic. In the future, we plan to extend our work to other discriminative tasks, for example, object detection and segmentation.

References

1. Samir Al-Stouhi and Chandan K Reddy. Transfer learning for class imbalance problems with inadequate data. *Knowledge and information systems*, 2016.
2. O. Mac Aodha, N. Campbell, J. Kautz, and G. J. Brostow. Hierarchical subquery evaluation for active learning on a graph. In *Conf. on Comput. Vis. Pattern Recognit.*, 2014.
3. A. Atzeni, M. Jansen, S. Ourselin, and J.E. Iglesias. A probabilistic model combining deep learning and multi-atlas segmentation for semi-automated labelling of histology. In *Conf. on Medical Image Comput. Comput. Assisted Intervention*, 2018.
4. W. Beluch, T. Genewein, A. Nurnberger, and J. Kohler. The power of ensembles for active learning in image classification. In *Conf. on Comput. Vis. Pattern Recognit.*, 2018.
5. Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-vae. *stat*, 2018.
6. Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011.
7. J. Donahue, J. Hoffman, E. Rodner, K. Saenko, and T. Darrell. Semi-Supervised Domain Adaptation with Instance Constraints. In *Conf. on Comput. Vis. Pattern Recognit.*, pages 668–675, 2013.
8. E. Elhamifar, G. Sapiro, A. Yang, and S. Shankar Sasrty. A convex optimization framework for active learning. In *Int. Conf. on Comput. Vis.*, 2013.
9. Y. Gal and Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Int. Conf. on Mach. Learn.*, pages 1050–1059, 2016.
10. Y. Gal, R. Islam, and Z. Ghahramani. Deep bayesian active learning with image data. In *Int. Conf. on Mach. Learn.*, 2017.
11. Y. Guo. Active instance sampling via matrix partition. In *Adv. Neural Inf. Process. Syst.*, 2010.
12. M. Hasan and A. K. Roy-Chowdhury. Context aware active learning of activity recognition models. In *Int. Conf. on Comput. Vis.*, 2015.
13. K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Conf. on Comput. Vis. Pattern Recognit.*, pages 770–778, 2016.
14. Yu He, Kechen Song, Qinggang Meng, and Yunhui Yan. An end-to-end steel surface defect detection approach via fusing multiple hierarchical features. *IEEE Transactions on Instrumentation and Measuremente*, 2019.
15. Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.
16. Y. Huo, Z. Xu, K. Aboud, P. Parvathaneni, S. Bao, C. Bermudez, S. Resnick, L.E. Cutting, and B.A. Landman. Spatially localized atlas network tiles enables 3d whole brain segmentation from limited data. In *Conf. on Medical Image Comput. Comput. Assisted Intervention*, 2018.
17. A. Joshi. Multi-class active learning for image classification. In *Conf. on Comput. Vis. Pattern Recognit.*, 2009.

18. D.P. Kingma and J. Ba. Adam: A Method for Stochastic Optimisation. In *Int. Conf. on Learn. Representations*, 2015.
19. Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, 2014.
20. D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *Int. Conf. on Learn. Representations*, 2014.
21. T.N. Kipf and M. Welling. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907*, 2016.
22. Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.
23. Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A Simple Unified Framework for Detecting Out-Of-Distribution Samples and Adversarial Attacks. In *Adv. Neural Inf. Process. Syst.*, pages 7167–7177, 2018.
24. D. D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. *Machine Learning Proceedings*, 1994.
25. Xin Li and Yuhong Guo. Adaptive active learning for image classification. In *Conf. on Comput. Vis. Pattern Recognit.*, 2013.
26. X. Li and Y. Guo. Multi-level adaptive active learning for scene classification. In *European Conf. on Comput. Vis.*, 2014.
27. Hui Lin, Bin Li, Xinggang Wang, Yufeng Shu, and Shuanglong Niu. Automated defect inspection of led chip using deep convolutional neural network. *Journal of Intelligent Manufacturing*, 2019.
28. B. Liu and V. Ferrari. Active learning for human pose estimation. In *Int. Conf. on Comput. Vis.*, 2017.
29. Zimo Liu, Jingya Wang, Shaogang Gong, Huchuan Lu, and Dacheng Tao. Deep reinforcement active learning for human-in-the-loop person re-identification. In *Int. Conf. on Comput. Vis.*, 2019.
30. W. Luo, A. Schwing, and R. Urtasun. Latent structured active learning. In *Adv. Neural Inf. Process. Syst.*, 2013.
31. A. K. McCallumzy and K. Nigamy. Employing em and pool-based active learning for text classification. In *Int. Conf. on Mach. Learn.*, 1998.
32. H. T. Nguyen and A. Smeulders. Active learning using preclustering. In *Int. Conf. on Mach. Learn.*, 2004.
33. Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
34. S. Paul, J. H. Bappy, and A. K. Roy-Chowdhury. The power of ensembles for active learning in image classification. In *Conf. on Comput. Vis. Pattern Recognit.*, 2017.
35. A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko. Semi-Supervised Learning with Ladder Networks. In *Adv. Neural Inf. Process. Syst.*, pages 3546–3554, 2015.
36. D. Roth and K. Small. Margin-based active learning for structured output spaces. In *European Conf. on Mach. Learn.*, 2006.
37. K. Saito, Y. Ushiku, and T. Harada. Asymmetric Tri-Training for Unsupervised Domain Adaptation. In *Int. Conf. on Mach. Learn.*, 2017.
38. O. Sener and S. Savarese. Active learning for convolutional neural networks: A core-set approach. In *Int. Conf. on Learn. Representations*, 2018.
39. Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.
40. B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. 2008.
41. Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *Int. Conf. on Comput. Vis.*, 2019.

42. Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, 2015.
43. S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2001.
44. S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. *Int. J. Comput. Vis.*, 2014.
45. Dan Wang and Yi Shang. Learning loss for active learning. In *Int. Joint Conf. on Neural Networks*, 2014.
46. H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo. Mind the Class Weight Bias: Weighted Maximum Mean Discrepancy for Unsupervised Domain Adaptation. In *Conf. on Comput. Vis. Pattern Recognit.*, pages 2272–2281, 2017.
47. Y. Yang, Z. Ma, F. Nie, X. Chang, and A. G. Hauptmann. Multi-class active learning by uncertainty sampling with diversity maximization. *Int. J. Comput. Vis.*, 2015.
48. D. Yoo and I. Kweon. A new active labeling method for deep learning. In *Conf. on Comput. Vis. Pattern Recognit.*, 2019.
49. Yong Jie Zhao, Yun Hui Yan, and Ke Chen Song. Vision-based automatic detection of steel surface defects in the cold rolling process: considering the influence of industrial liquids and surface textures. *The International Journal of Advanced Manufacturing Technology*, 2017.
50. Z. Zhou, J. Shin, L. Zhang, S. Gurudu, M. Gotway, and J. Liang. The power of ensembles for active learning in image classification. In *Conf. on Comput. Vis. Pattern Recognit.*, 2017.
51. Tuanfei Zhu, Yaping Lin, and Yonghe Liu. Synthetic minority oversampling technique for multiclass imbalance problems. *Pattern Recognition*, 2017.

Appendix

A Detailed results

Here, we show results that were omitted from the main paper due to spatial constraints, including the raw experimental results.

A.1 With a different classifier

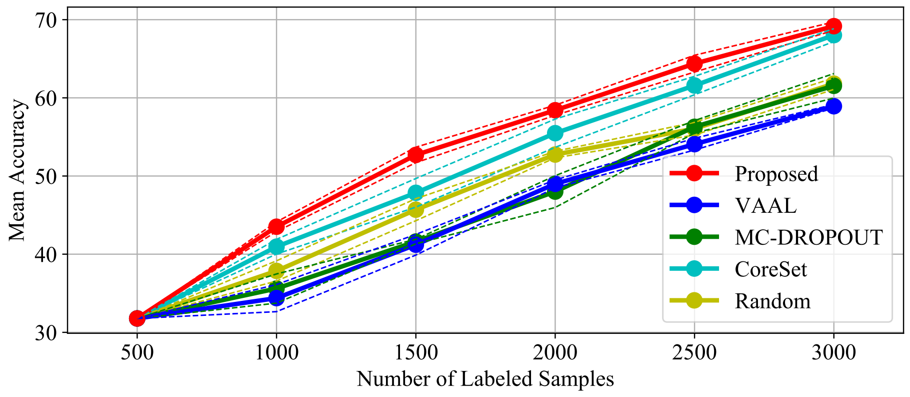
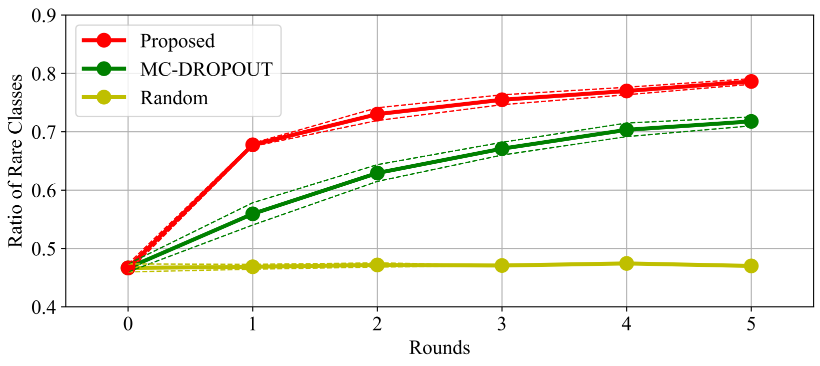


Fig. 8. Results for CIFAR-10$^{+[1]}$ with ResNet-34

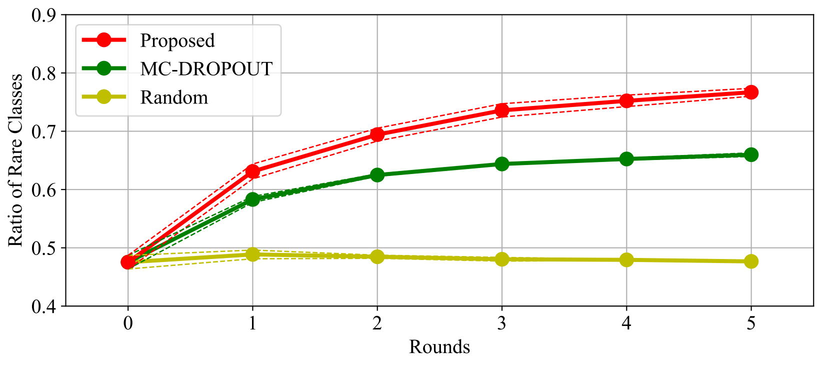
Our method also works well with more complex networks – ResNet-34. In Fig. 8, we replace the classifier network with ResNet-34 and repeat CIFAR-10$^{+[1]}$ experiment in Section 4. As shown, our method outperforms the state of the art even with a different classifier network.

A.2 How much are we focusing on rare classes?

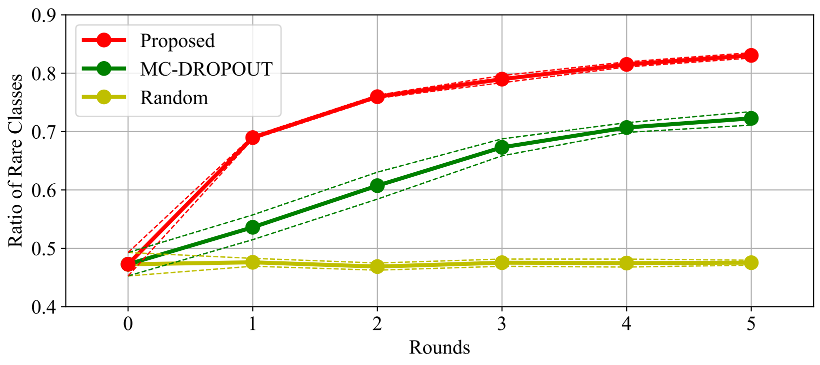
When rare classes are present, our method focuses on those, as the prior probability drives towards a balanced training set. In Fig. 9, we report the ratio of rare classes in the labelled pool for CIFAR-10$^{+[1]}$, CIFAR-10$^{+[5]}$, and CIFAR-10$^{+[10]}$. As shown, the ratio of rare classes increase, showing that our method indeed creates a balanced training set, which eventually leads to a better classifier as reported in Section 4. Note that MC-DROPOUT also tend to deliver similar results, but not as fast as our method. As a result, this contributes to the performance improvement in terms of classification accuracy.



(a) CIFAR-10$^{+[1]}$



(b) CIFAR-10$^{+[5]}$



(c) CIFAR-10$^{+[10]}$

Fig. 9. Sample ratio of the rare classes for dominant datasets

A.3 Detailed results with various budget sizes

In Fig. 10, we present the entire results of comparisons with various budget sizes. Since smaller budget size tends to allow methods to react faster to the newly labelled training samples, the final performance of each method conventionally reduces as its budget size increases. According to the results, our method outperforms all the compared methods even with various budget sizes.

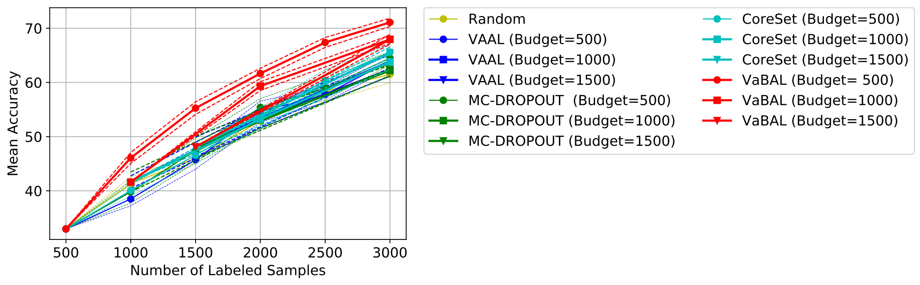


Fig. 10. Results of ablation tests with various budget sizes on *dominant* CIFAR-10.

A.4 Detailed results with various λ value

In Fig. 11, we report the performance of our method with various λ including $\lambda = 0$ that was omitted in Fig. 7(b). In contrast to the outperforming performance with other λ values, when $\lambda = 0$ the performance dramatically drops. Since $\lambda = 0$ means that the class-wise condition in the latent features of VAE is entirely ignored, it becomes infeasible to correctly estimate the three probability terms in Eq. 6 by using the VAE. Therefore, the samples to be labelled at every stage are extracted without considering the label prediction, which results in worse performance even than the *Random* scheme.

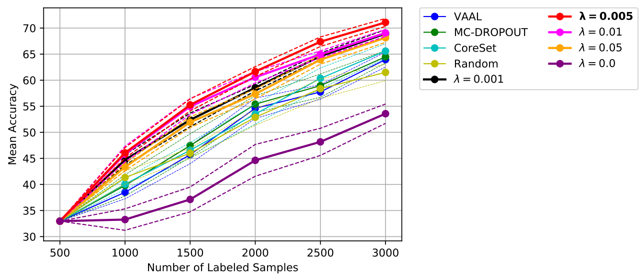
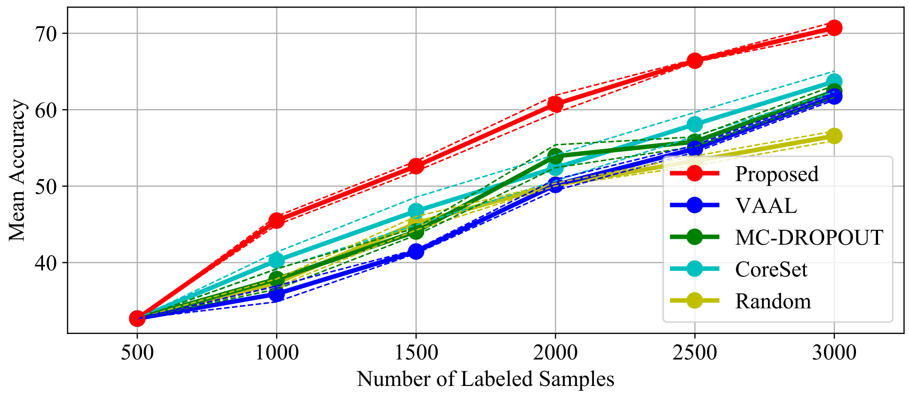


Fig. 11. Results with varying λ on *dominant* CIFAR-10.

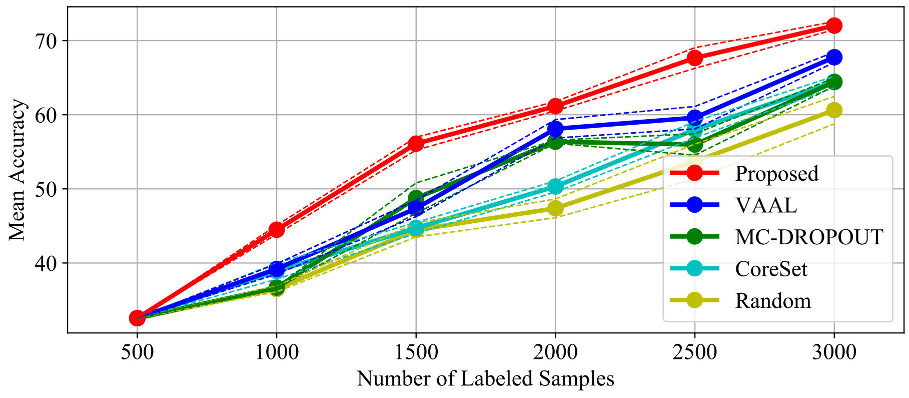
B Individual results

As mentioned in the main script, we report the individual result to demonstrate that our results are not by chance. In Fig. 12 we show the averages of the three trials for each dataset of CIFAR-10$^{+[1]}$, CIFAR-10$^{+[5]}$, and CIFAR-10$^{+[10]}$. Likewise, in Fig. 13 we present the plots averaging the three trials for CIFAR-10$^{-[1]}$, CIFAR-10$^{-[5]}$, and CIFAR-10$^{-[10]}$.

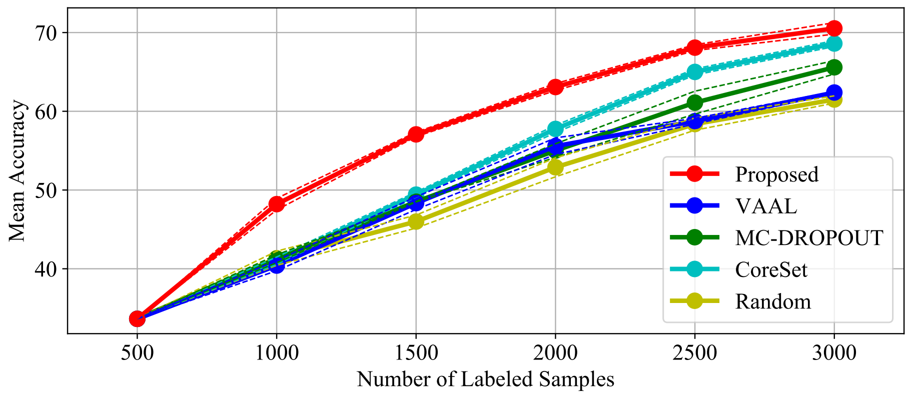
Finally, in Figs. 14$-$ 22 we provide all the individual results of our framework before being averaged. For competing methods, we plot the aggregated results for all three trials, so that one can easily compare with the maximum possible result of the competitors. As show, our results always outperform the competition.



(a) Results for CIFAR-10$^{+[1]}$

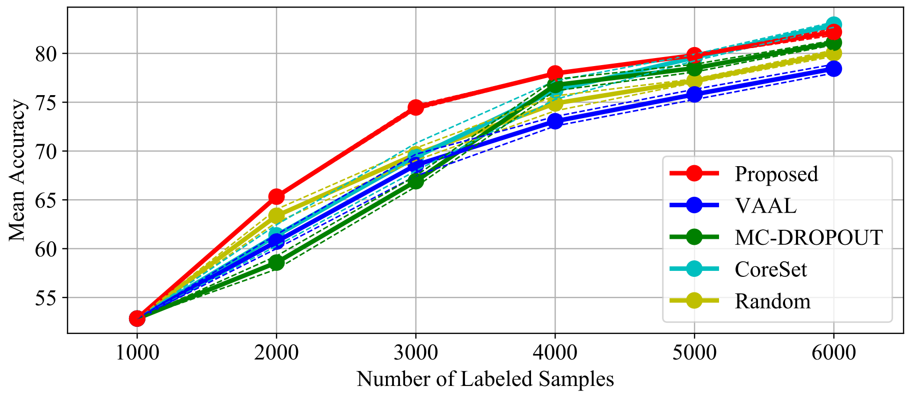


(b) Results for CIFAR-10$^{+[5]}$

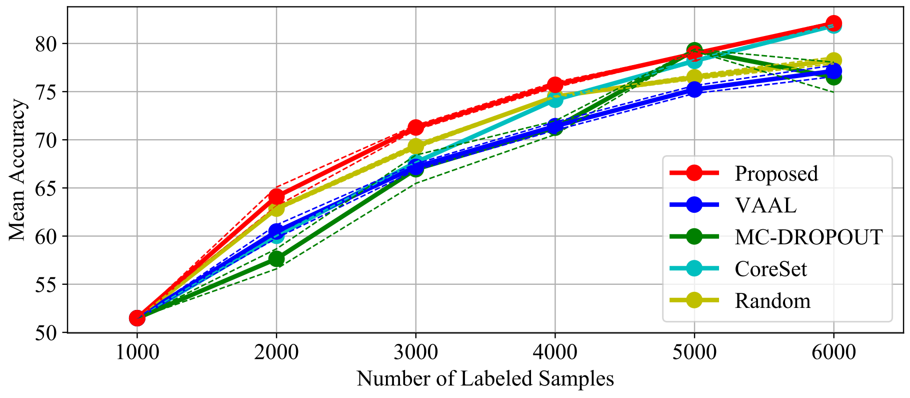


(c) Results for CIFAR-10$^{+[10]}$

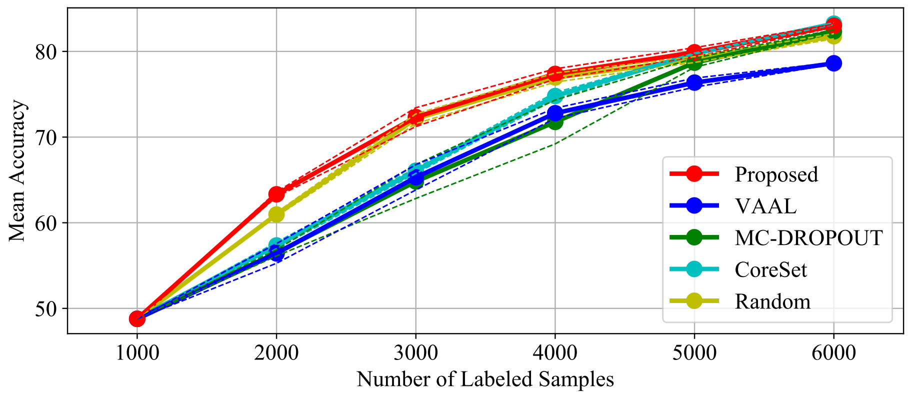
Fig. 12. Results for the dominant variants of CIFAR-10



(a) Results for CIFAR-10$^{-[1]}$

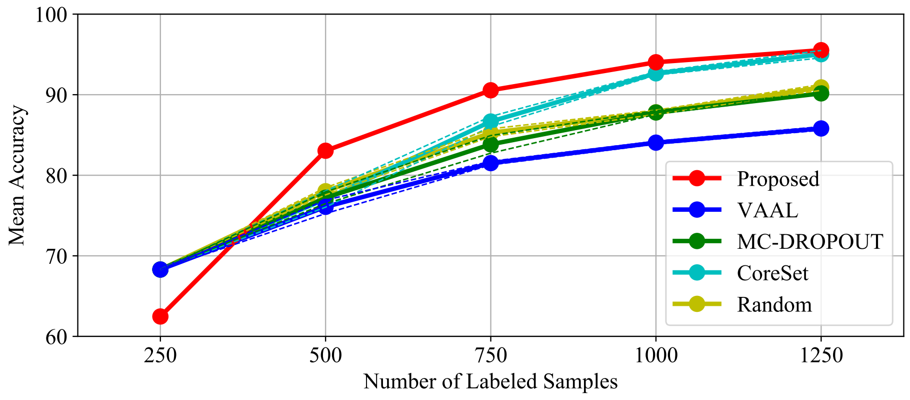


(b) Results for CIFAR-10$^{-[5]}$

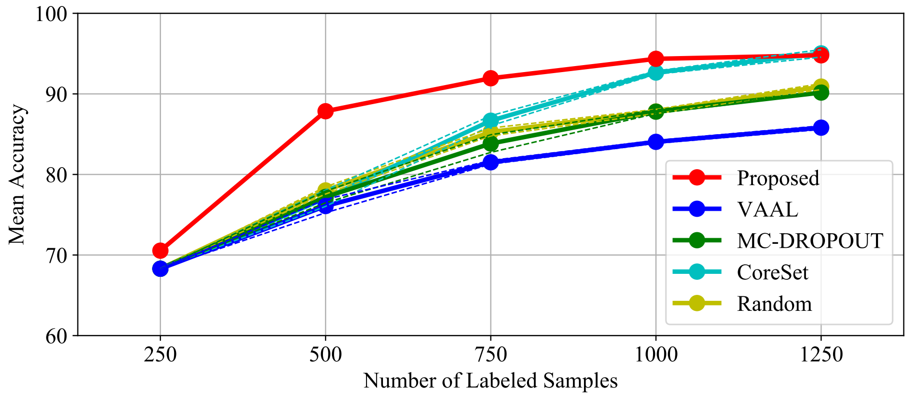


(c) Results for CIFAR-10$^{-[10]}$

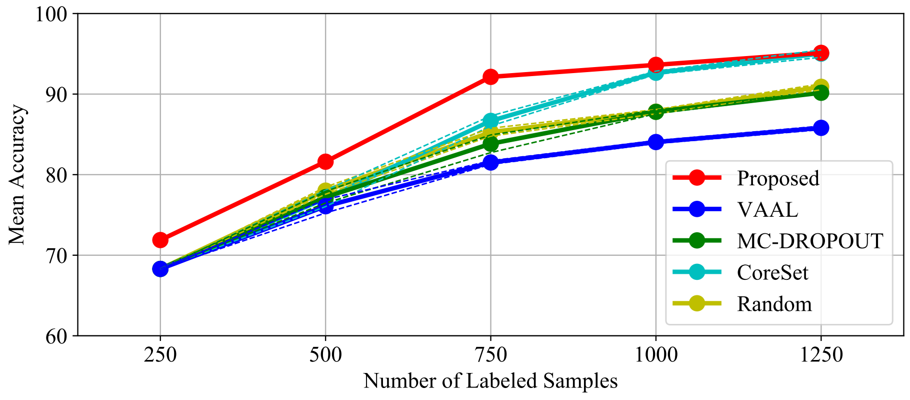
Fig. 13. Results for the rare variants of CIFAR-10



(a) Trial 1

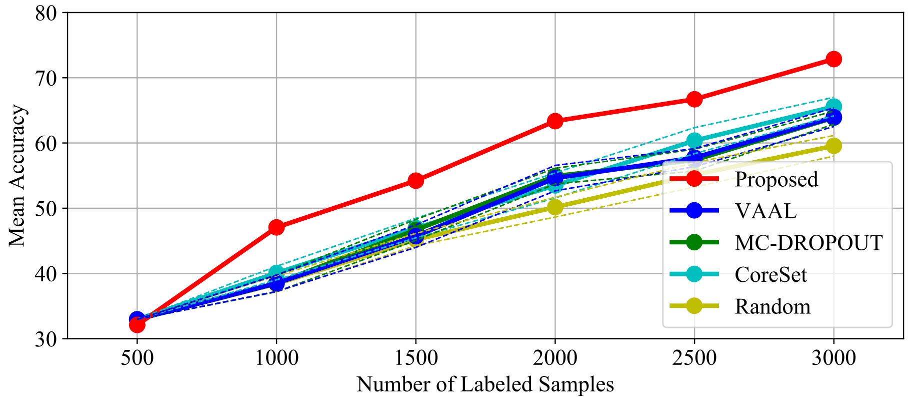


(b) Trial 2

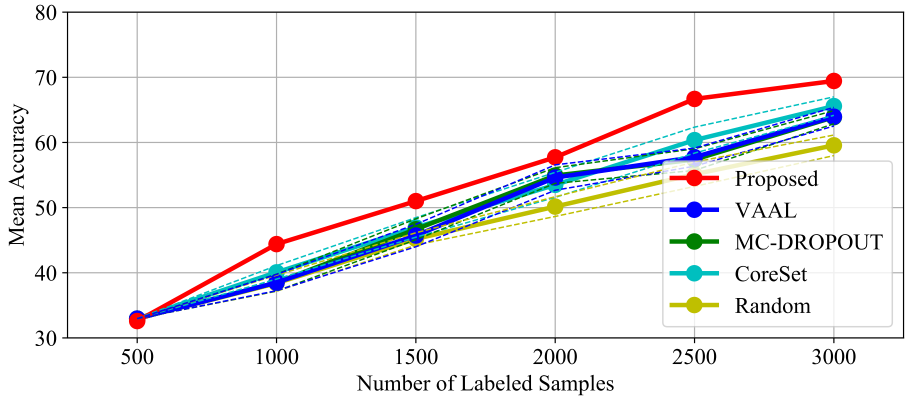


(c) Trial 3

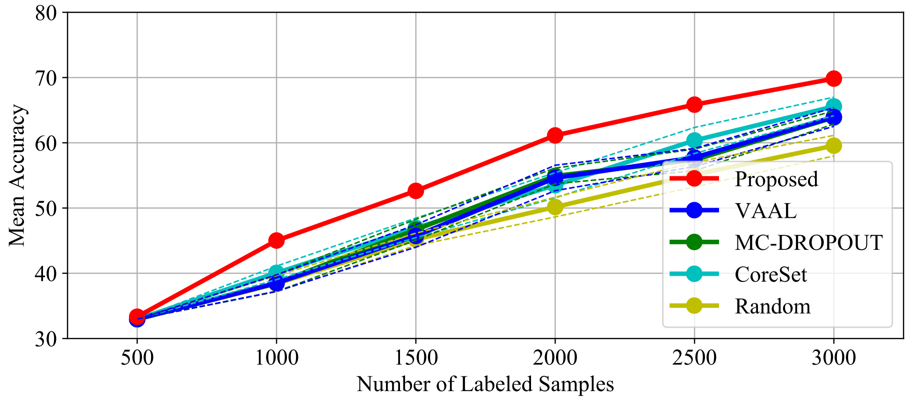
Fig. 14. Individual results for the NEU dataset



(a) Trial 1

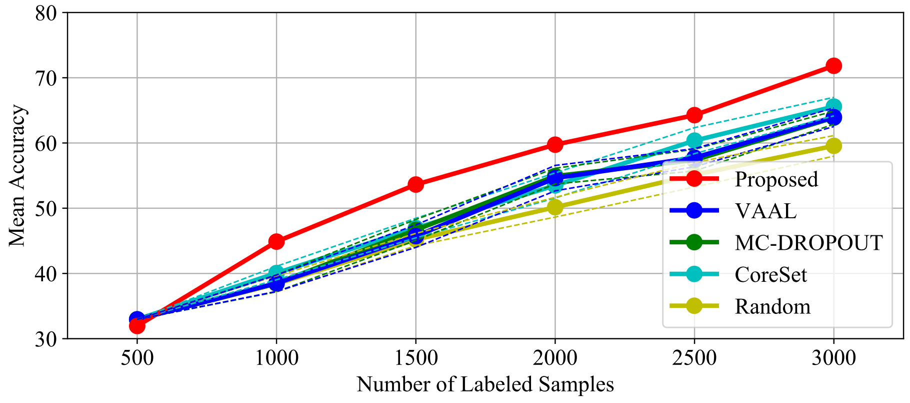


(b) Trial 2

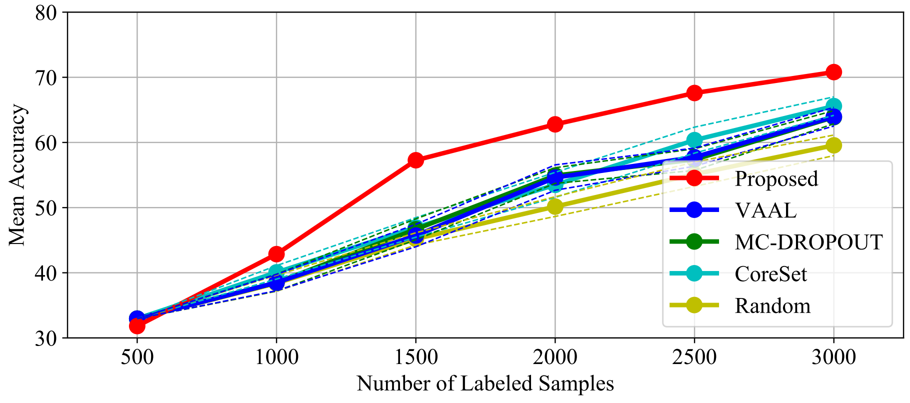


(c) Trial 3

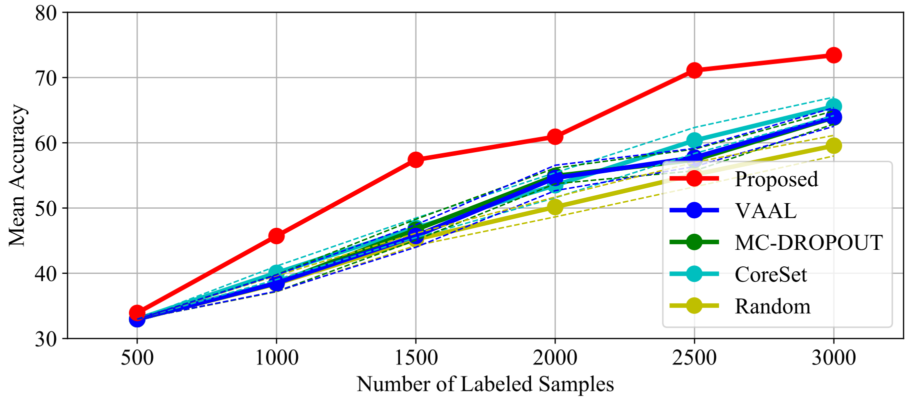
Fig. 15. Individual results for CIFAR-10$^{+[1]}$



(a) Trial 1

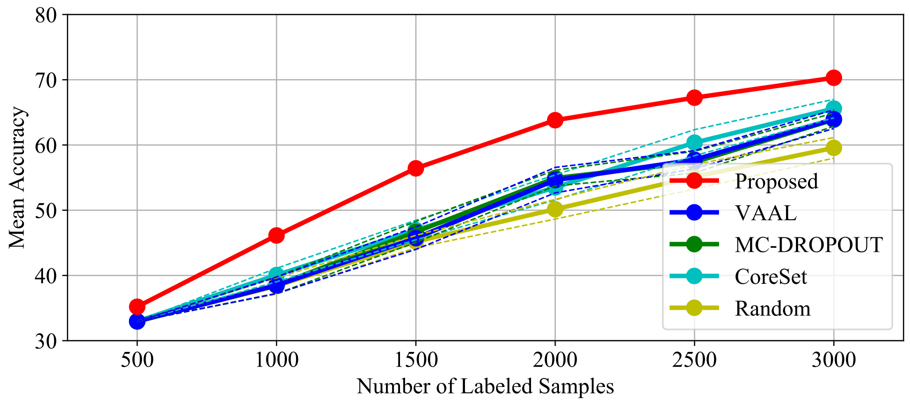


(b) Trial 2

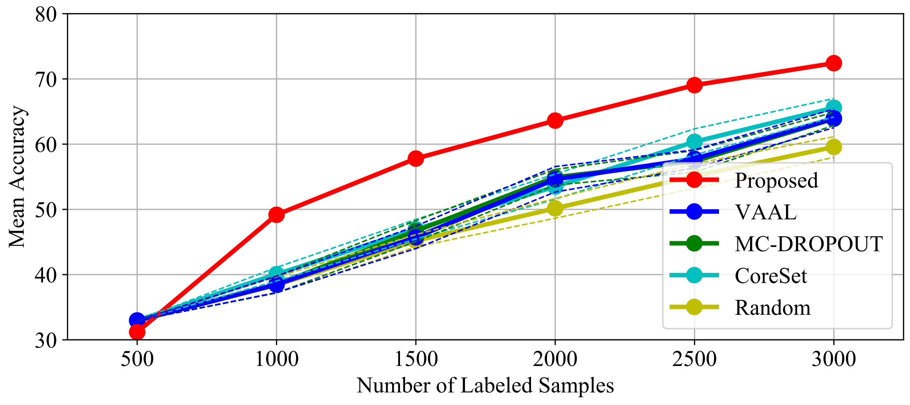


(c) Trial 3

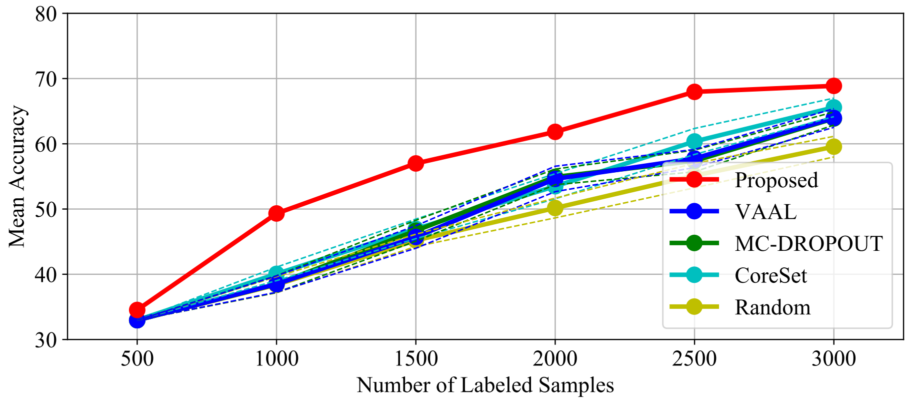
Fig. 16. Individual results for CIFAR-10$^{+[5]}$



(a) Trial 1

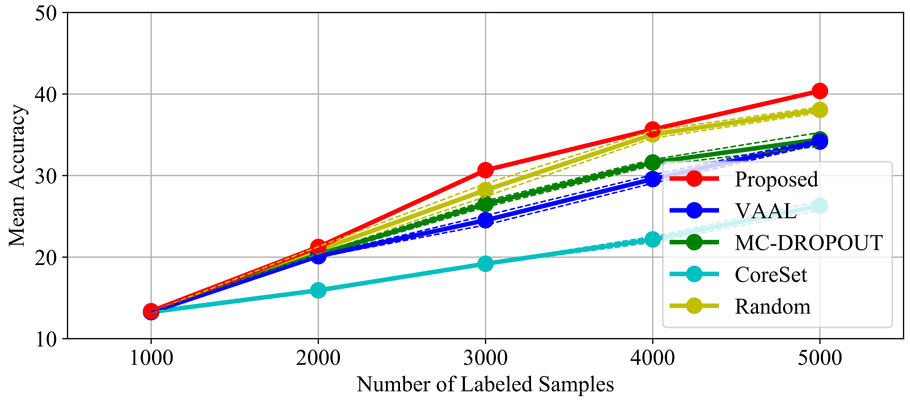


(b) Trial 2

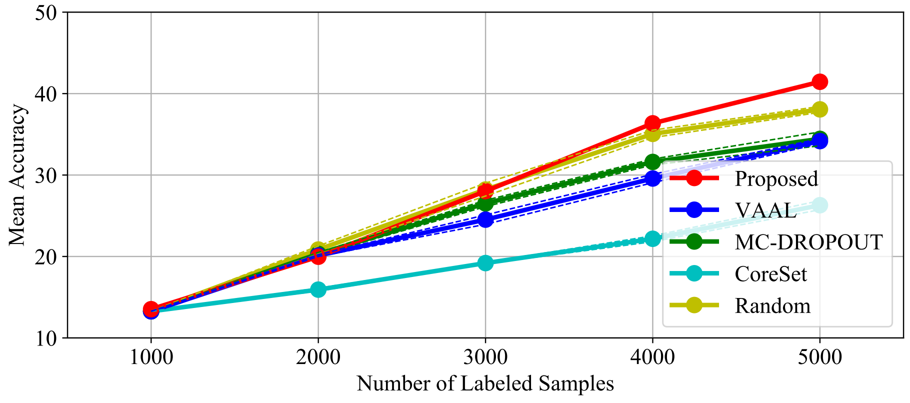


(c) Trial 3

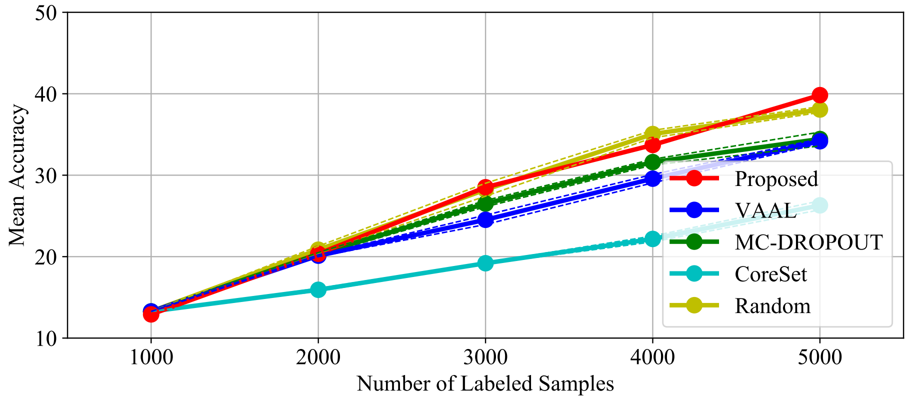
Fig. 17. Individual results for CIFAR-10$^{+[10]}$



(a) Trial 1

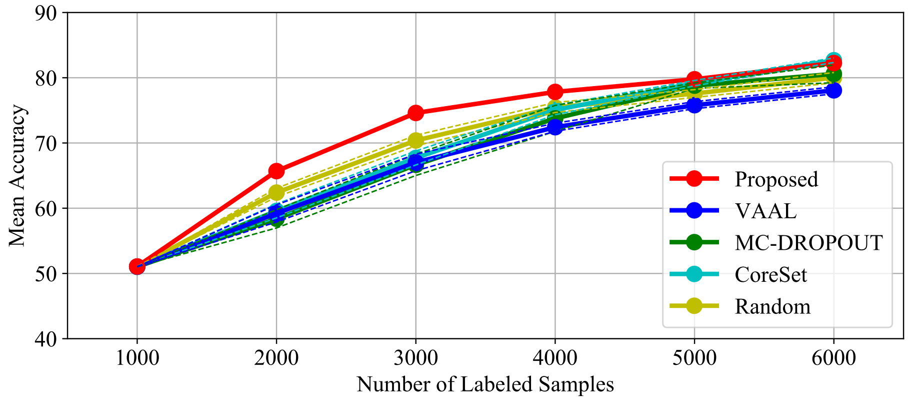


(b) Trial 2

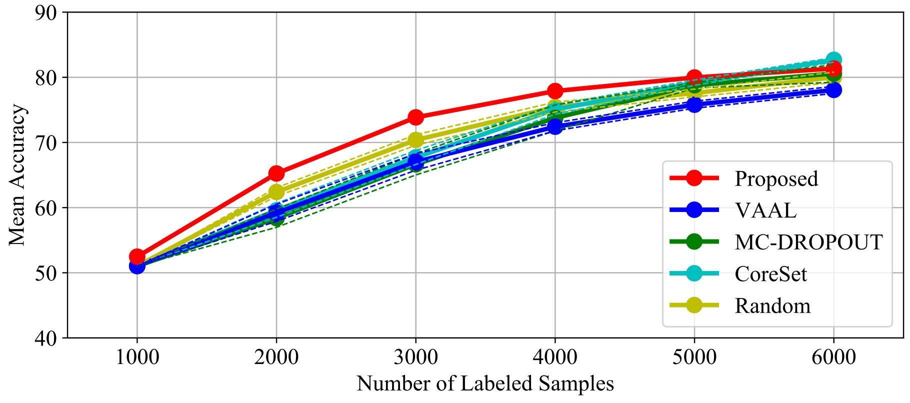


(c) Trial 3

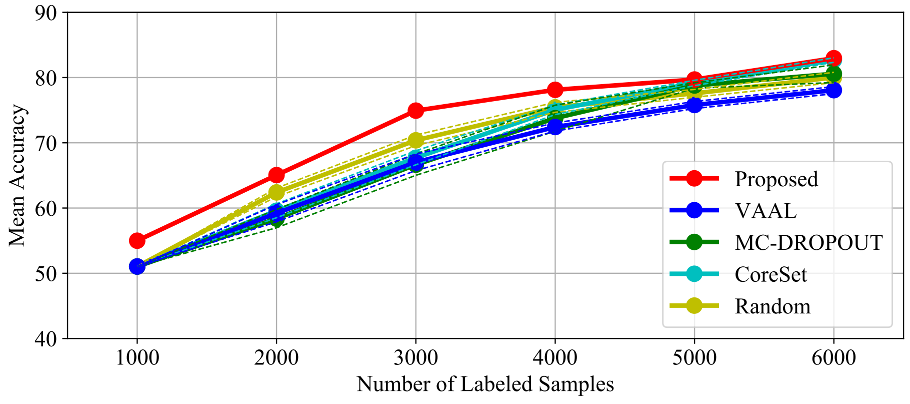
Fig. 18. Individual results for CIFAR-100$^{+[45:55]}$



(a) Trial 1

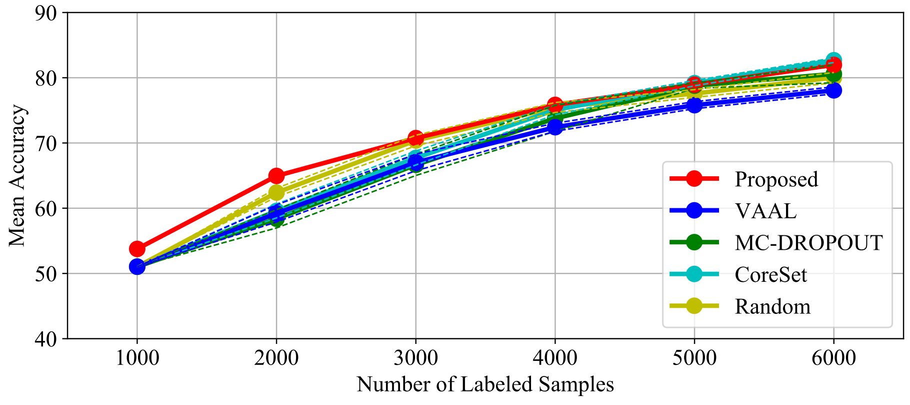


(b) Trial 2

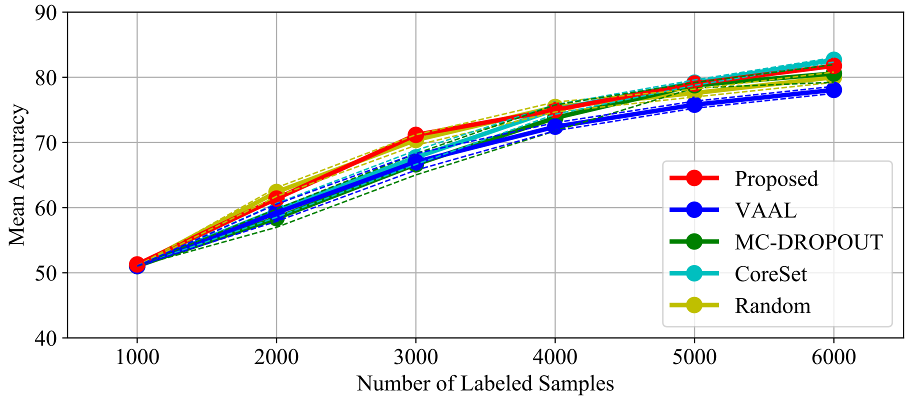


(c) Trial 3

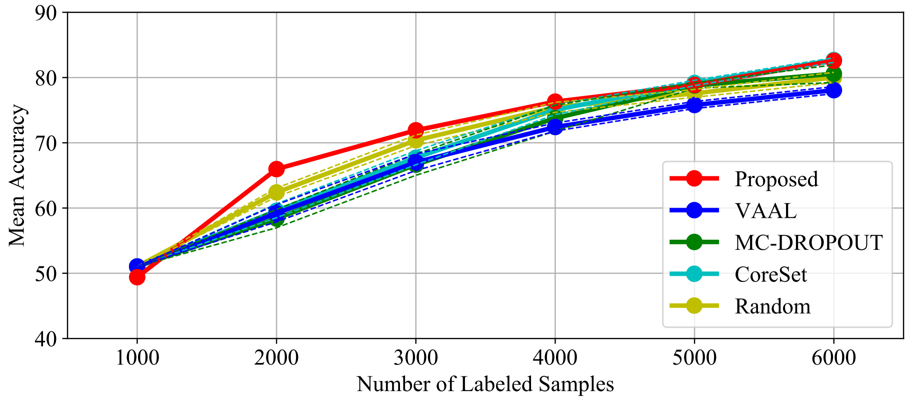
Fig. 19. Individual results for CIFAR-10$^{-[1]}$



(a) Trial 1

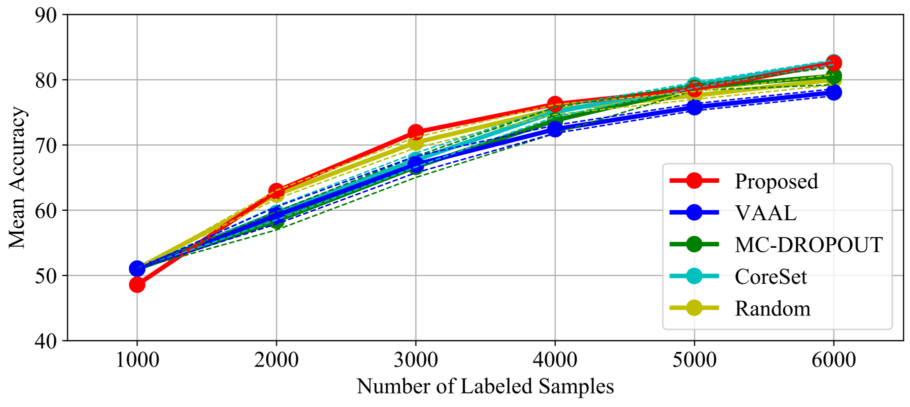


(b) Trial 2

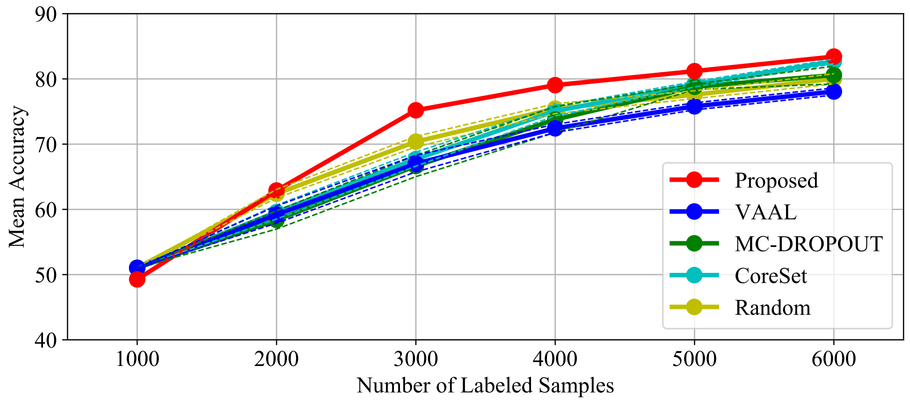


(c) Trial 3

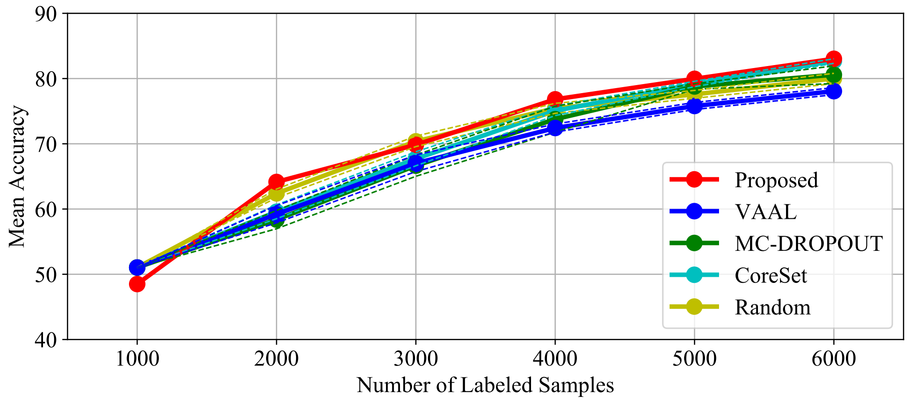
Fig. 20. Individual results for CIFAR-10$^{-[5]}$



(a) Trial 1

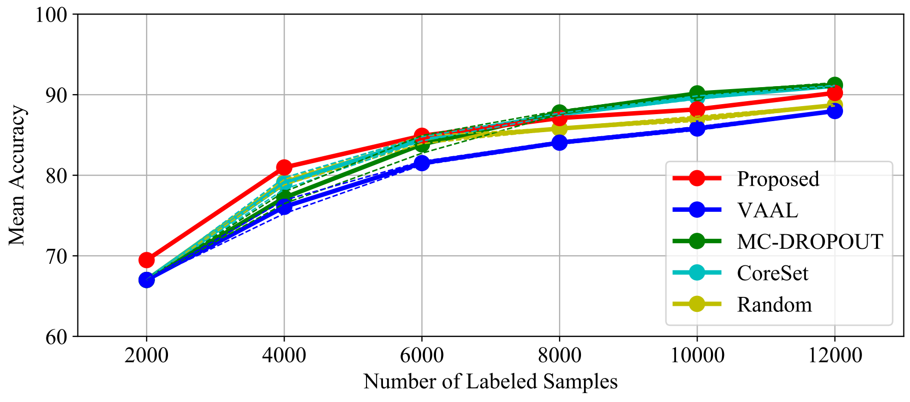


(b) Trial 2

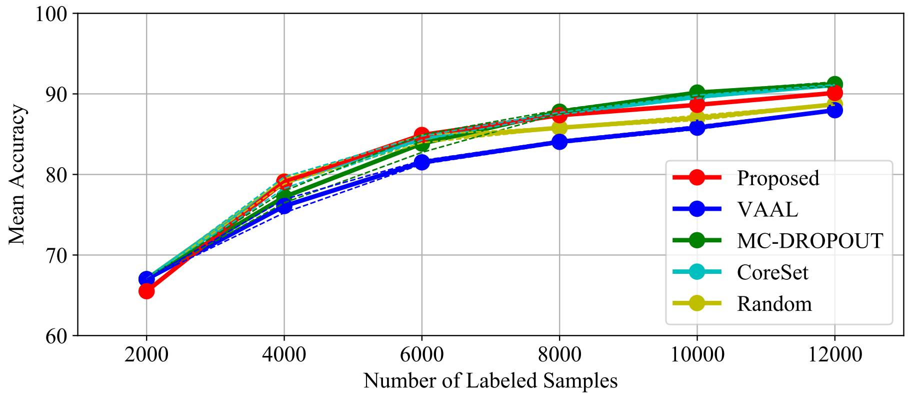


(c) Trial 3

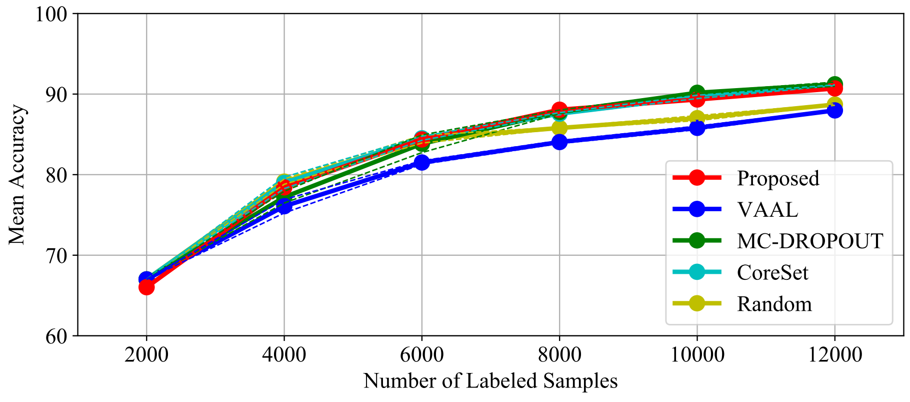
Fig. 21. Individual results for CIFAR-10$^{-[10]}$



(a) Trial 1

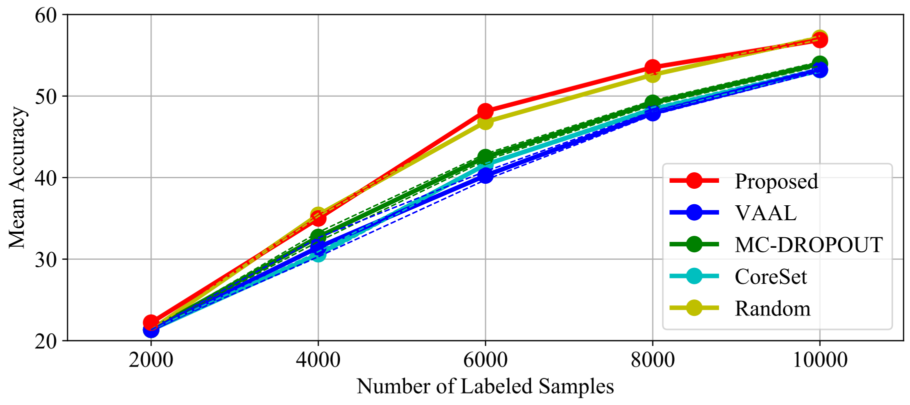


(b) Trial 2

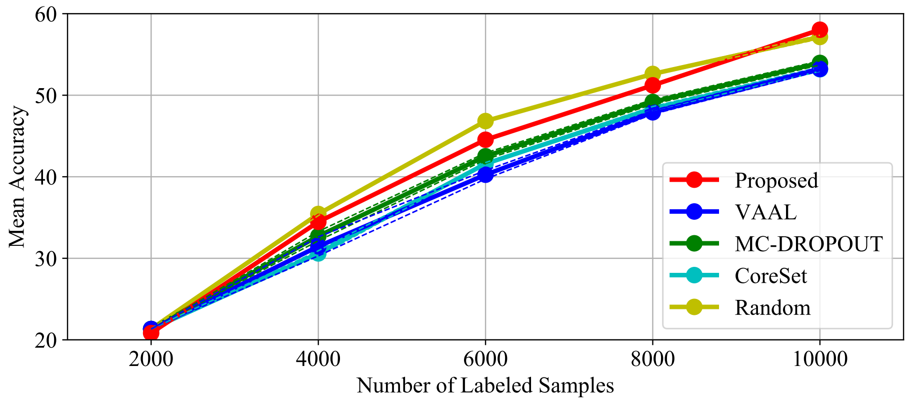


(c) Trial 3

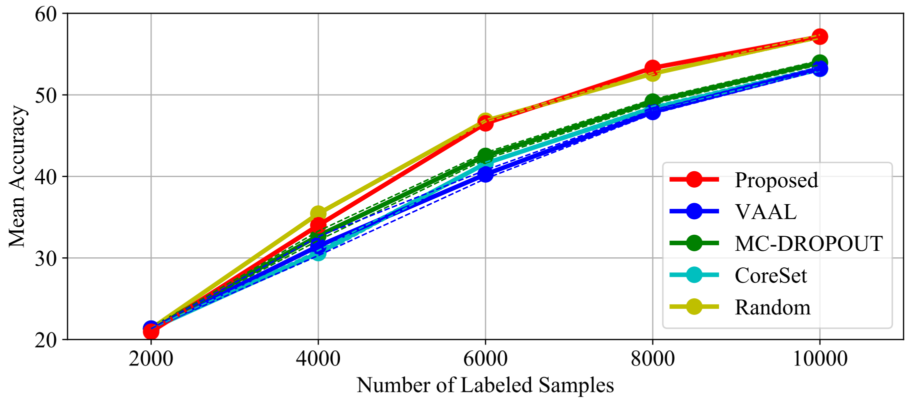
Fig. 22. Individual results for the full CIFAR-10 dataset



(a) Trial 1



(b) Trial 2



(c) Trial 3

Fig. 23. Individual results for the full CIFAR-100 dataset