Efficiently correcting machine learning: considering the role of example ordering in human-in-the-loop training of image classification models

Geoff Holmes
geoffrey.holmes@waikato.ac.nz
University of Waikato
Hamilton, New Zealand

Eibe Frank
eibe.frank@waikato.ac.nz
University of Waikato
Hamilton, New Zealand

Dale Fletcher
dale.fletcher@waikato.ac.nz
University of Waikato
Hamilton, New Zealand

Corey Sterling
corey.sterling@waikato.ac.nz
University of Waikato
Hamilton, New Zealand

ABSTRACT

Arguably the most popular application task in artificial intelligence is image classification using transfer learning. Transfer learning enables models pre-trained on general classes of images, available in large numbers, to be refined for a specific application. This enables domain experts with their own—generally, substantially smaller—collections of images to build deep learning models. The good performance of such models poses the question of whether it is possible to further reduce the effort required to label training data by adopting a human-in-the-loop interface that presents the expert with the current predictions of the model on a new batch of data and only requires *correction* of these predictions—rather than *de novo* labelling by the expert—before retraining the model on the extended data. This paper looks at how to order the data in this iterative training scheme to achieve the highest model performance while minimising the effort needed to correct misclassified examples. Experiments are conducted involving five methods of ordering, using four image classification datasets, and three popular pre-trained models. Two of the methods we consider order the examples *a priori* whereas the other three employ an active learning approach where the ordering is updated iteratively after each new batch of data and retraining of the model. The main finding is that it is important to consider accuracy of the model in relation to the number of corrections that are required: using accuracy in relation to the number of labelled training examples—as is common practice in the literature—can be misleading. More specifically, active methods require more cumulative corrections than *a priori* methods for a given level of accuracy. Within their groups, active and *a priori* methods perform similarly. Preliminary evidence is provided that suggests that for "simple" problems, i.e., those involving fewer examples and classes, no method improves upon random selection of examples. For more complex problems, an *a priori* strategy based on a greedy sample selection method known as "kernel herding" performs best.

IUI '22, March 22–25, 2022, Helsinki, Finland

ACM ISBN 978-1-4503-9144-3/22/03…$15.00
https://doi.org/10.1145/3490099.3511110

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**.

KEYWORDS

Convolutional Neural Networks, Image Classification, AI Interfaces

ACM Reference Format:
Geoff Holmes, Eibe Frank, Dale Fletcher, and Corey Sterling. 2022. Efficiently correcting machine learning: considering the role of example ordering in human-in-the-loop training of image classification models. In *27th International Conference on Intelligent User Interfaces (IUI '22), March 22–25, 2022, Helsinki, Finland*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3490099.3511110

1 INTRODUCTION

Deep learning technology has revolutionized artificial intelligence to the point where many tasks involving images, speech, and text can be readily modelled using neural networks. The particular application we consider in this paper is image classification. High-performance network architectures for this problem have been developed, but good classification accuracy in a new problem domain hinges on the availability of sufficiently large amounts of labelled data that has been classified by domain experts. As expert time can be very costly, there is a strong incentive to reduce the amount of labelled training data that is required to achieve an acceptable level of accuracy. The arguably most successful method to achieve this is to transfer knowledge that has been learned from another, sufficiently similar, image classification problem. In particular, there is a large collection of publicly available image classification models that have been trained on the well-known ILSVRC2012 ImageNet dataset containing 1,200,000 labelled images, each belonging to one of 1,000 categories of real-world objects. Any network in this collection can be adapted to a new target domain by replacing its last layer with one that has the appropriate size for the classification problem at hand. Crucially, when this network is trained on the data from the target domain, only the parameters of this last layer are initialised randomly; the remaining parameters are those obtained from ImageNet. This yields a much better starting point

to learn appropriate parameters for the target domain than training a network from scratch—which is normally performed by setting all parameters of all network layers to random values initially.

Applying this form of transfer learning based on "fine-tuning" pre-trained models is a very effective and widely applied method to reduce the amount of labelled data required to successfully train a model in the target domain. However, the labelling effort required to obtain a sufficient amount of data for fine-tuning can still be substantial. Additionally, some labelled testing data must be acquired to validate the fine-tuned network by establishing an estimate of predictive accuracy. The latter must be representative and unbiased; thus, the data labelled for testing is normally chosen randomly from the set of unlabelled data instances. A key observation that has given rise to a wide variety of so-called "active" learning methods is that this requirement does not hold for the training data: "anything goes" in attempts to obtain a set of training examples that are as informative as possible and obtain a model of the greatest possible accuracy. Consequently, there is a large number of heuristic sample selection criteria in the literature of active learning, many of which show promise in reducing the amount of training data that must be labelled for a given level of accuracy.

Active learning places the human expert in the loop when building a machine learning model: the training set is collated iteratively, using the model trained on the data that has been labelled so far to inform the choice of further examples to be labelled by the expert. When used with neural networks, labelling is performed in batches. An initial batch of data is chosen randomly and labelled by the expert, the pre-trained model is fine-tuned on this data, and a new unlabelled batch of data is chosen for labelling—those examples that are deemed most informative based on the model learned so far and whichever sample selection heuristic is used to measure informativeness. These steps are performed iteratively until a sufficient level of accuracy is achieved on the test data.

In this paper, we examine a new measure of active learning effectiveness that is more relevant to practical applications, and demonstrate empirically that it can yield significantly different conclusions regarding the relative performance of active learning with respect to simple random sampling. Our key insight is that it is not appropriate to simply count the number of examples that must be labelled by human experts when evaluating sampling strategies because (a) the model available in the current iteration of human-in-the-loop learning can (and should) be exploited to "pre"-label examples and (b) simply correcting the examples mislabelled by the model requires much less human effort than labelling the examples from scratch without considering the models' guesses.

We make one assumption, namely that a human expert with a batch of data, labelled by the current model, will only have to make an effort on examples that are misclassified. We believe that this is a reasonable assumption in image classification problems involving natural objects because all the images in a batch, arranged according to their corresponding predicted labels, can be presented to the expert on a single screen, and the human visual system has evolved to easily pick out anomalies in a collection of images.

We present experiments with four image classification datasets and three popular convolutional neural network architectures pre-trained on ImageNet. In these experiments, we evaluate batch active learning using entropy-based sample selection, a standard strategy in active learning, and batch active learning with stochastic acquisition, a recent variant specifically proposed for active learning using deep neural networks. We compare active learning to simple random sampling and a method known as "kernel herding", which greedily picks examples to yield more representative training data than random sampling. This is achieved by using a similarity function with certain mathematical properties known as a "kernel function". We also consider a new active learning variant of kernel herding that updates the similarity function after every update of the neural network model based on the latest batch of labelled data. This paper makes the following contributions:

- We introduce the notion of the number of corrections during training as a performance metric, and show that it yields different conclusions than simply counting the number of examples that must be labelled.
- We conduct empirical studies using the metric to determine the most effective approach for human-in-the-loop machine learning.
- We examine a variety of practical methods for active and non-active sample selection.
- We discuss the implications of our results when incorporating sample selection in human-in-the-loop systems using transfer learning for image classification.

2 RELATED WORK

In the scenario considered in this paper, once an initial model has been built, machine learning algorithm performance, sample selection strategy, and difficulty of the application task combine to determine how many iterations are required in the human-in-the-loop system when it is deployed in practice. We focus on deep learning using neural networks, but it worth noting that with the exception of streaming algorithms [1], the majority of machine learning algorithms are batch-trained so our findings may translate to other settings where identifying misclassifications places a low cognitive load on the human expert.

In almost all current practical applications of machine learning, model training either occurs as a one-off, once all training examples have been labelled, or the model is regularly retrained on the most recent data. In either mode, labels are needed and human input is required to obtain them.

Providing class labels for a classification problem is a simple form of data annotation. An interface offering AI assistance for this task is presented in [4], considering text classification as the problem domain. The focus is on using AI to identify the most probable labels and reducing the set of labels that the human annotator must choose from. The paper goes beyond basic binary classification—as we do here—and explores how semi-supervised learning [21] can assist the process. A deep sentence encoder is used in conjunction with a centroid-based AI model. Results demonstrate that labelling speed and accuracy can be improved with assistance. The labelling task, related to short-text customer inquiries, has a focus on non-specialists providing the labels.

Approaches to human-in-the-loop deep learning where time complexity and application task difficulty are both high are starting to emerge [11–13, 16] particularly where specialist expertise for

correcting model errors is crucial, as in medical image analysis. Active learning is explored to select the best samples for optimizing model performance as well as exploring model output interactions and scaling to deployment level [2]. For this specialist application, the authors examine how human and machine can co-develop models by gathering data, modelling, and then iteratively refining the model towards higher performance. As a future direction, the authors consider transfer learning as a promising approach in the general area of medical image analysis [20] noting that pre-trained models using natural images that are refined using medical images can outperform models trained from scratch on only medical images, which is consistent with the scenario we consider in this paper.

A recent survey of human-in-the-loop and machine learning can be found in [23]. The main application area covered is natural language processing which, due to its wide applicability, is an ideal domain for human-in-the-loop research and development. In the area of computer vision, the survey highlights a growing number of papers in the areas of object detection, and image restoration, segmentation, and enhancement. While image classification is not surveyed directly, the "exciting questions" in [23] resonate with some of the goals of this paper: how to select key examples, and how to establish an evaluation benchmark.

Deep learning methods in human-centred machine learning are surveyed in [8], which covers several wide-ranging application domains. While the survey is written for non-AI-experts, it is focused on specialists within those domains.

An interesting approach to the iterative co-development of machine learning models is provided by [14]. Interactive machine learning attempts to leverage both human domain expertise and a human's ability to teach to improve model performance. Of particular relevance to this research on image classification is the question of "seeing" things differently (human and model). We do not explore this question here, but it is important, so offer some ideas for future work on this topic in the conclusions.

The use of explainability methods for image classification using deep networks is starting to emerge. In the context of this study, however, there are at least two issues to be considered: first, the particular computational method used for generating saliency (heat) maps (there is no consensus in the literature on deep learning on which one is most accurate) and, second, how to best present this information in an actual user interface. Once these issues have been resolved, the associated technologies would clearly make a difference to human-in-the-loop model development.

3 ORDERING EXAMPLES

We examine five approaches to ordering examples in the experiments presented in this paper: two methods that establish an order *a priori*—before machine learning is invoked—that remains static throughout the human-in-the-loop learning process, and three methods that implement a form of active learning, where the next batch of selected examples is chosen based on what machine learning has inferred from previously observed expert feedback. The two different types of approaches we consider are illustrated in Figure 1, which shows the user-in-the-loop process for the *a priori* and active learning methods. In both methodologies, the training

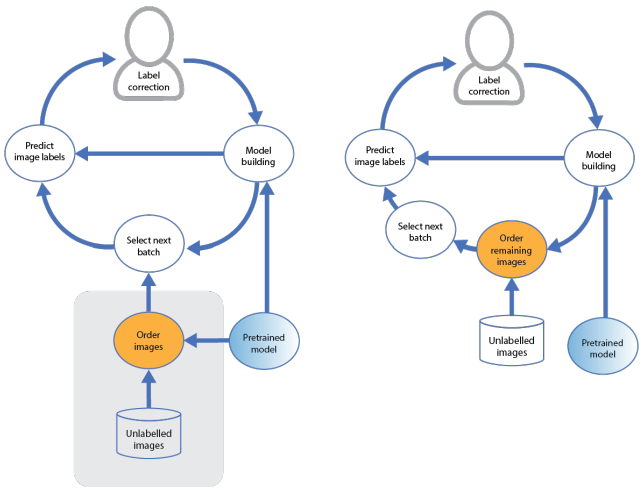

Figure 1: Two ways of ordering examples for human-in-the-loop learning: *a priori* (left) and using active learning (right).

loop entails presenting the user with a new selection of images that have been pre-labelled according to the predictions given by the current state of the classification model. The user corrects any misclassifications, and this batch of curated examples is then added to the training set for the next iteration of the model.

The difference between the methodologies lies in the mechanism for the ordering used in selecting the next batch of images, as depicted in orange. In the *a priori* methods, shown on the left, the ordering of the images is outside of the loop. The ordering is done once, so that within the loop, the selection of the next batch of images is simply the next images in line, according to that initial ordering.

In the active learning methods, shown on the right, the ordering of the images is inside the loop. These methods can take advantage of extra information, such as the evolving—and presumably improving—embeddings from the classification model, and the distribution of these embeddings over the remaining unlabelled images. Next, we describe the two *a priori* approaches.

3.1 Random sampling

The first ordering method we evaluate is random sampling without replacement: to sample one example from the remaining pool of unlabelled examples, we pick one at random with equal probability; once picked, an example is no longer available for selection.

Random sampling is a simple baseline strategy that is unbiased. Moreover, the reservoir sampling algorithm [22] can be employed to yield a random sample from a data stream where the total number of data points is not known initially. A disadvantage of the random sampling approach is that there is no guarantee any particular subset of randomly chosen examples is representative of the full

set of unlabelled examples: due to the nature of random selection, parts of the full set may be over- or under-represented.

3.2 Kernel Herding

A deterministic alternative to random sampling is kernel herding [3], which can be used to greedily construct a subsample of a dataset in an intelligent manner so that it provides better coverage of the full data. Moreover, because kernel herding constructs the subsample by selecting one example at a time, it also provides an order of the examples that we can use for human-in-the-loop learning.

The kernel herding algorithm for sample selection is quite straightforward: greedily select examples from the full dataset, starting with the one that has the greatest average similarity to all the examples in the full dataset (i.e., the single example that represents the distribution of the full dataset "best"); then, in each subsequent iteration, pick the example that represents the full dataset well while being as dissimilar as possible to the instances that have been picked in previous iterations of the greedy process.

The name "kernel herding" is used because similarity between examples is measured using a so-called "kernel function", a similarity function that has certain mathematical properties—more precisely, it represents an inner product in an implicitly-defined feature space. Assume X is the full unlabelled dataset from which to select instances and $k(\vec{x}, \vec{y})$ is some kernel function that measures the similarity between two instances x and y. The kernel herding selection rule applied to greedily select examples is

$$\vec{x}_{T+1} = \arg\max_{\vec{x} \in X} \left(\left(\frac{1}{|X|} \sum_{\vec{y} \in X} k(\vec{x}, \vec{y}) \right) - \frac{1}{T+1} \sum_{t=1}^{T} k(\vec{x}, \vec{x}_t) \right), \quad (1)$$

where T is the number of samples that have already been selected.

Note that when the first example is picked, i.e., when $T = 0$, the second sum in the equation has value zero. This means only the first sum is relevant, and the algorithm simply picks the example $\vec{x}_1$ with the greatest average similarity to all examples in the full pool X. In subsequent iterations, the second sum, which iterates over all examples $\vec{x}_1, ..., \vec{x}_T$ that have already been selected, comes into play, and encourages the selection of a new example x_T that has low average similarity to previous selections while also exhibiting high average similarity to the full set.

Applying kernel herding requires choosing an an appropriate kernel function to measure similarity between examples. In our experiments, we apply the Gaussian radial basis function kernel:

$$k(\vec{x}, \vec{y}) = e^{-\frac{||\vec{x}-\vec{y}||^2}{2\sigma^2}}. \quad (2)$$

It is widely used in machine learning to train kernel-based models such as support vector machines and has the appealing theoretical property of being a characteristic kernel [19]. The parameter of the Gaussian radial basis function is chosen to be 0.01, which makes $\sigma^2 = 50$ in the equation above. This is the default value used in the WEKA machine learning software [5]. No optimisation of this parameter was attempted.

In the scenario considered in this paper, the individual examples are images. Applying a basic kernel function to measure the similarity of images by using their bitmaps as feature vectors $\vec{x}$ and $\vec{y}$ does not yield meaningful results in general. Embeddings provided by convolutional neural networks can be used instead to yield the feature vectors $\vec{x}$ and $\vec{y}$ of two images being compared. In our experiments with basic kernel herding, we use convolutional neural networks pre-trained on ImageNet to compute these embeddings. Given a particular image as input, the output of the penultimate layer of the network is used as the feature vector (i.e., the output of the layer preceding the output layer of the network).

3.3 Active Learning using Uncertainty Sampling

The previous ordering methods order examples *a priori*, before the iterative human-in-the-loop learning process begins (the left-hand diagram in Figure 1). Now, we proceed to discuss example selection methods that are placed into the loop, using the most recently built machine learning model to decide which examples are chosen to be presented to the human expert (the right-hand diagram in Figure 1). Methods for selecting samples in this manner are called "active learning methods" [17]. Generally, they attempt to select examples that will maximally improve the predictive performance of the model once they are included in the training set.

The first method we consider is uncertainty sampling [10], a classic approach to active learning. It is based on the idea that examples should be chosen for labelling for which the model is most uncertain when predicting a label. Machine learning models such as convolutional neural networks for image classification trained using the standard softmax activation function are able to produce a categorical probability distribution providing, for each image being classified, an estimated probability for each of the class labels. This categorical probability distribution can be used to obtain a measure of predictive uncertainty. In our experiments, we use the entropy of this distribution to quantify uncertainty.

When applying this type of active learning in the standard setting considered in the literature, the initial set of samples for training the first model is chosen randomly because estimates of uncertainty are not yet available. Subsequently, examples are ranked by entropy, based on the latest model available, and the highest-ranked batch of examples is forwarded to the human expert for labelling.[1] However, it is important to reiterate here that the scenario considered in this paper is different from what is commonly assumed when using and evaluating active learning approaches: in our scenario, in addition to the unlabelled examples themselves, the current model's predicted class labels are also forwarded to the human expert. Once the expert has corrected the labels, the labelled data is added to the training set constructed in the previous iterations of the human-in-the-loop learning process, and the model is retrained on the expanded training data.

3.4 Batch Active Learning with Stochastic Acquisition

When used to select a batch of examples, uncertainty sampling and similar strategies that employ an acquisition function such as entropy to compute scores for individual examples are potentially problematic because they do not provide a mechanism to promote diversity of the examples that are selected: if the model is very

[1] Computational complexity renders purely incremental active learning infeasible when using deep neural networks; thus, batches of data must be used.

uncertain regarding the classification of particular image in the unlabelled set, and another very similar image exists in this set, it will also exhibit high entropy, and both will be selected into the batch presented to the human expert, which will be inefficient.

A very recent proposal is to introduce a stochastic component into the example selection mechanism used by batch active learning to inject diversity into the batch of examples selected for labelling [9]. The proposed method defines a probability distribution over the unlabelled examples and takes a sample based on this distribution to form a batch to be labelled. Assume $a(\vec{x})$ is the value of an acquisition function such as entropy for the unlabelled example $\vec{x}$. The probability $p(\vec{x}_i)$ of picking a particular example $\vec{x}_i$ to be included in the current batch is then defined as

$$p(\vec{x}_i) = \frac{e^{a(\vec{x}_i)/t}}{\sum_j e^{a(\vec{x}_j)/t}}, \tag{3}$$

where t is a so-called "temperature parameter" that determines how close the distribution is to uniform. Increasing the value of t will bring the probabilities for different examples $\vec{x}_i$ closer together and encourage greater diversity when these probabilities are used for sampling. The most appropriate value of t depends on the choice of acquisition function. For entropy calculated using base-2 logarithms, with the acquisition function we use, we found $t = 0.125$ to provide a good trade-off between example diversity and exploitation of what has already been learned by the model.

3.5 Active Learning using Kernel Herding

A third active learning strategy we consider in this paper is a simple adaptation of kernel herding. As discussed above, when applying this method *a priori*, before human-in-the-loop learning starts, we use embeddings obtained from a convolutional neural network pre-trained on ImageNet to represent images for the purpose of applying the kernel function. However, once labelled examples in the target domain are available, the convolutional feature extractor is fine-tuned on the labelled data, providing a potentially more suitable model for extracting embeddings that can be used in the kernel. In other words, whenever a retrained, updated model becomes available during the human-in-the-loop learning process, this model should be used to produce embeddings of the remaining unlabelled data available for selection and kernel herding should be applied to pick examples based on these feature vectors.

Although this strategy has the advantage of exploiting information that has already been learned from the labelled data obtained for the target domain so far, it also bears potential disadvantages: (a) all the remaining unlabelled data must be re-embedded based on the latest model—and the kernel function must be recomputed for all new pairs of feature vectors—and (b) the feature space into which the data is embedded is biased in the sense that the model has been fitted to pull apart, in this space, groups of data from the target domain belonging to different classes. The impact of this on the behaviour of kernel herding is difficult to quantify.

4 EXPERIMENTS

Our experiments are designed to be reproducible by using publicly available image collections, publicly available pre-trained models, and by making code available to implement the ordering methods[2]. We start by describing the datasets and the rationale for the choice of pre-trained models, then explain the experimental design and the degree of difficulty of modelling each dataset.

4.1 Dataset Descriptions

We used the four publicly available datasets for the experiments that are listed in Table 1. They cover a fairly wide range of collection sizes (from 1700 to 20,000), class counts (20 to 200) and modelling difficulty (easy to hard). The smallest dataset is the American Sign Language dataset[3] and contains between 50 and 90 images per label. Each image is a hand producing the sign of a single letter of the English alphabet. The hand is located in the same part of each image, under fairly constant lighting conditions, making this dataset quite uniform.

The Flowers dataset[4] contains an irregular class distribution, with some classes containing approximately 40 images, others containing over a 100, and one containing 251. Each image consists of a fairly centrally located flower, but there is significant variation in the types of background.

The Dog Breeds dataset[5] has 120 classes containing 100 between 200 images each, classified by breed. The position of the dog as well as the background vary enormously in this dataset.

The Birds dataset[6] has 200 categories with most containing 59 or 60 images each, but a small number of categories contain fewer, with the smallest class containing 41. Each image is of a single bird, classified by species, in a similar fashion to the Dogs dataset. The position of the bird in the image is typically central, although the background does vary a great deal from image to image.

4.2 Methodology

Our experiments are designed to mimic behaviour when performing human-in-the-loop training in practice. The implementation is based on Python, making use of the popular Keras (https://keras.io/) deep-learning library for the pre-trained models [6]. Three models are chosen from those made available within the Keras package. Mobilenet is used because it is popular when developing a light-weight solution for a problem, in particular for mobile applications. The other two pre-trained models we use are both deep Residual Networks [7] (ResNets) chosen to represent medium-weight and heavyweight solutions with two size options of a 50 layer deep network and one with 152 layers. The environment for utilising the GPU (an Nvidia GeForce GTX 1080Ti) was controlled via Docker images.

The experimental process applied in our experiments can be summarised as follows: first, a stratified holdout dataset is generated, consisting of 15% of the images from each class in a given dataset, selected at random. This dataset is used to evaluate the current model at each iteration of the training loop. The remaining 85% are considered the training dataset and are ordered according

[2] https://github.com/waikato-ufdl/iui2022
[3] https://public.roboflow.com/object-detection/american-sign-language-letters
[4] https://www.robots.ox.ac.uk/ vgg/data/flowers/102/
[5] http://vision.stanford.edu/aditya86/ImageNetDogs/main.html
[6] http://www.vision.caltech.edu/visipedia/CUB-200-2011.html

Table 1: Datasets

Name	Number of Examples	Number of Classes
American Sign Language	1728	26
Flowers	8189	102
Stanford Dog Breeds	20,580	120
Birds	11,788	200

to the particular experiment's ordering approach.[7] Next, the chosen pre-trained model is iteratively fine-tuned with successively larger portions of the training dataset, increasing by 50 examples per iteration. For the *a priori* methods, these examples are simply the next 50 items as determined by the pre-ordering of the training dataset. For the active learning methods, the remaining items in the training dataset are re-ordered using the method being evaluated before selecting the 50 most relevant items according to that method.

Optimization using gradient descent generally employs a validation set that is used for early stopping. Hence, once the training dataset for the next iteration of training has been assembled by adding the 50 chosen examples, a randomly-selected 15% is removed for internal validation. The remaining training items are randomly shuffled and used to fine-tune the pre-trained model using gradient descent.

The initial weights for the models used in this process are those obtained by training on ImageNet, and fine-tuning is performed using the sparse categorical cross-entropy loss function, the ADAM optimiser with an initial learning rate of 0.0001 and a decay of 0.000001, and accuracy as the only metric for early stopping. In each iteration, fine-tuning is applied for a maximum of 100 epochs of mini-batch stochastic gradient descent with 5 images at a time.

After the fine-tuning step, the model is evaluated against the holdout dataset to establish an estimate of predictive accuracy of the model at this point in the simulated human-in-the-loop training process. However, once the 50 additional items for the next iteration have been selected, the current model is also evaluated against these 50 examples, as a measure of how well it performs when pre-labelling those items. This allows us to establish the number of examples that would need to be corrected by the human involved.[8]

5 RESULTS

We present our results as a collection of plots presented in four figures. Each individual plots has five graphs corresponding to the five ordering methods we evaluate. There is a matrix of 12 plots in each figure. Each row in a matrix has three plots corresponding to the three network architectures used, and there are four rows, one for each dataset. The three columns are ordered by the complexity of the network architecture (layer depth) increasing from left-to-right.

[7]Note that in the two active learning approaches that apply entropy, the entropy scores will provide a random order at this stage because the head of the network will have random initial weights before fine-tuning stats on the labelled data from the target domain.

[8]Because all the benchmark data used in our experiments is fully labelled, we can simply compare the ground-truth labels and the predicted labels in our simulated human-in-the-loop set-up.

6 DISCUSSION

The first set of plots, in Figure 2, contains standard learning curves showing classification accuracy on the holdout data as the number of labelled training instances increases. In each plot, the x-axis shows the number of training iterations. As explained above, for each iteration, an additional 50 training images are used to build the model. Classification accuracy of the model on the holdout data is on the y-axis. The plots show that as expected, accuracy of the trained models increases as more labelled data becomes available for training, regardless of the method that is used to order the data. Moreover, the accuracy of all methods appears to converge to approximately the same point as the number of trained examples increases towards the size of the full dataset. Considering accuracy earlier in the learning process, there are noticeable differences between ordering methods when applying the more sophisticated residual network architectures: on Dog breeds and Birds, *a priori* kernel herding yields better results than all the other methods, which all behave very similarly; on Flowers, *a priori* kernel herding outperforms the other ordering methods very early in the learning process but is then overtaken by its active variant, and the other two active learning methods are also competitive. In contrast, it is not possible to identify any meaningful differences between ordering methods on the comparatively small sign language dataset. Also, considering the simple Mobilenet architecture, there appears to be little difference in the results for the five ordering methods, excluding Flowers, where the embeddings used by the *a priori* kernel herding approach clearly yield a suboptimal sample ordering at the intermediate stage of the learning process.

The next set of plots, in Figure 3, exhibits much more substantial differences between ordering methods. Here, rather than looking at the amount of labelled data used for training the model, we consider the number of corrections that are necessary when, in each iteration of the human-in-the-loop process, the current version of the model is used to pre-label examples in the next batch of data. As argued above, this more accurately reflects the human effort required during this process. In each iteration, the next batch of images to add for training is selected using the given ordering method. Then, the current model is used to predict the label for each image, and the user will only need to correct and relabel those images for which the predicted label is incorrect. In each iteration, the number of corrections can range from zero (100% successful model prediction) to 50 (0% success).

The plots show a consistent effect across all datasets and pre-trained models. The active and *a priori* methods present as two distinct groups. Very clearly, the latter methods generally require fewer corrections per iteration than the active learning approaches. At or near convergence of classification accuracy (cf. Figure 2),

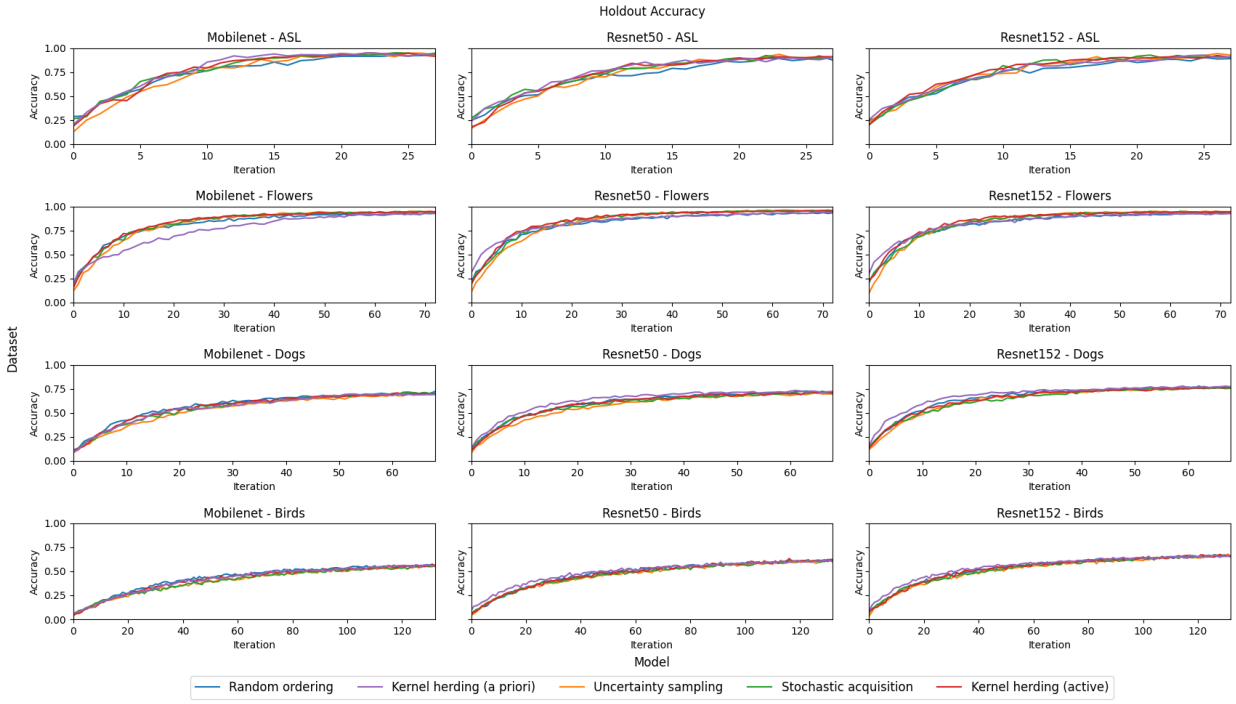

Figure 2: Holdout accuracy

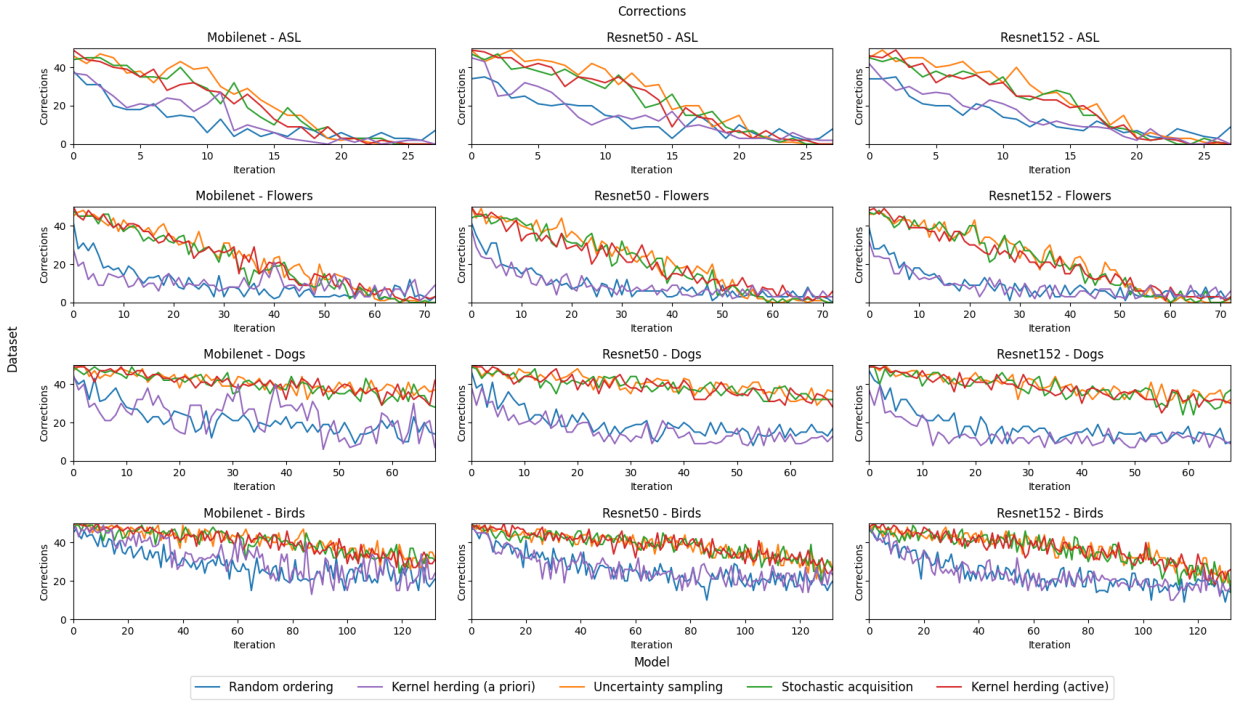

Figure 3: Corrections needed per iteration

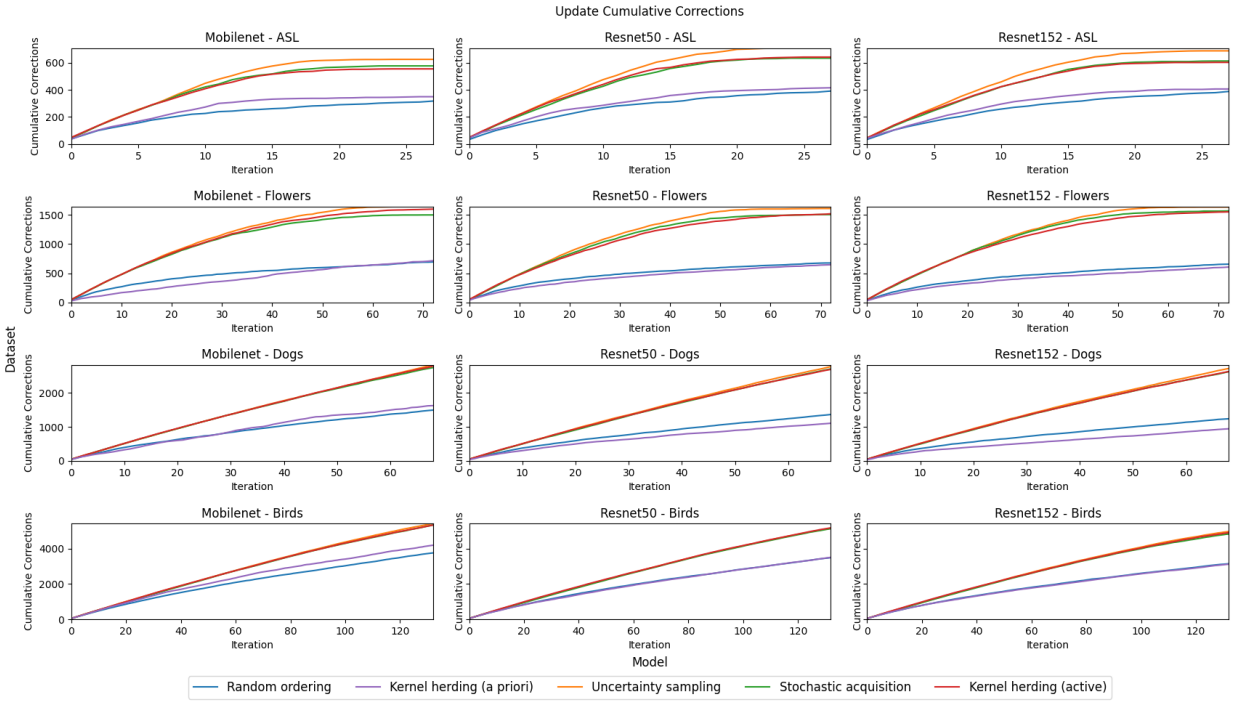

Figure 4: Cumulative correction counts

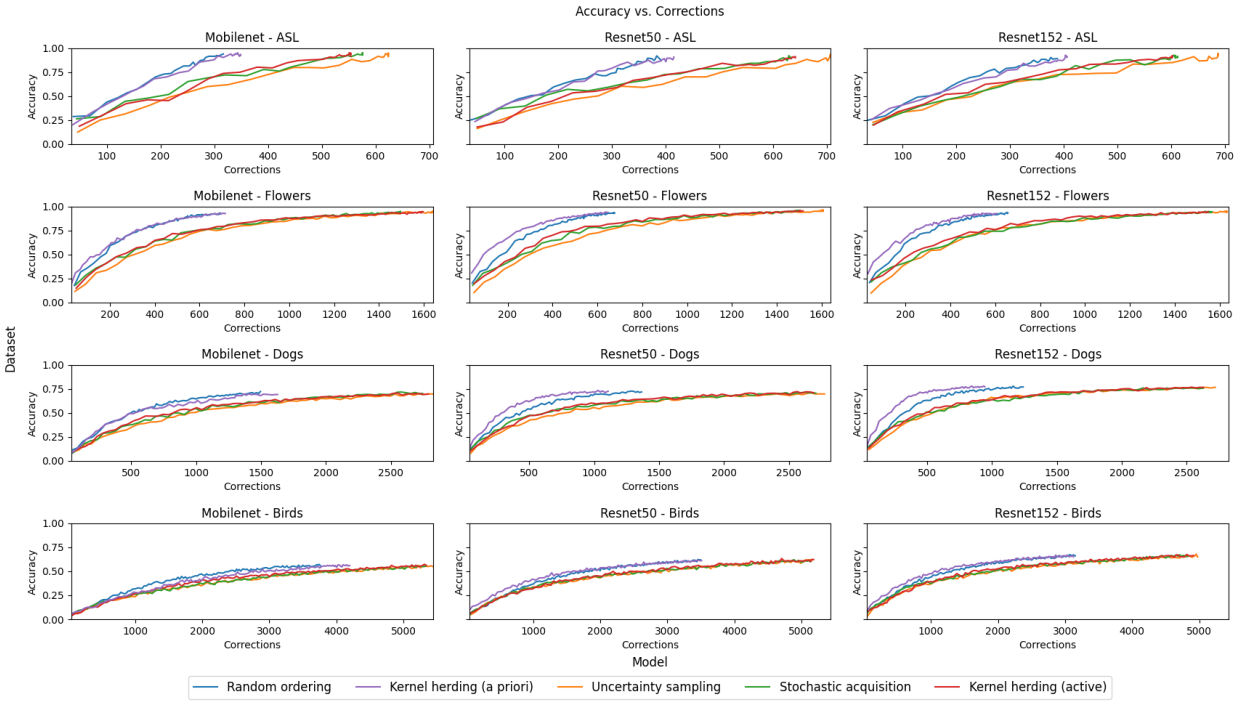

Figure 5: Accuracy versus number of corrections

active learning requires about as many corrections as the other methods on three of the datasets, but it incurs a noticeably larger cost at earlier iterations of the learning process. Moreover, considering the fourth dataset (Flowers), the gap in the number of required corrections remains even when accuracy has largely converged.

This behaviour seems surprising at first but can be explained by considering that active learning methods are designed to identify examples that maximize the amount of new information that is incorporated during training. For example, when applying the two methods that employ entropy-based selection, we explicitly favour those examples for which the current model is uncertain and not confident in assigning the correct class label. Thus, when the model is subsequently used to pre-label these examples, their labels are more likely to be incorrect, requiring human intervention to correct them.

The results show that this phenomenon is not restricted to the entropy-based active learning methods: the behaviour of "active" kernel herding is very similar, in spite of the fact that it does not explicitly take uncertainty of the model into account. Clearly, the fact that the embedding produced by the model has been adapted to the target domain is sufficient to force kernel herding into sampling images that are more likely to be incorrectly classified by this model.

This behaviour also explains why the difference between active learning and the *a priori* methods in terms of the number of required corrections disappears once the human-in-the-loop process has progressed sufficiently: early in the process, there is a large number of unlabelled examples for which the model is uncertain, but it becomes harder to find such difficult examples when the pool of unlabelled examples becomes smaller and the model more accurate. By this stage in the iterative process, the active learning methods have already selected the most 'difficult' examples and are left with images that are well-represented in the current model.

To provide a more direct view of the cost incurred by different ordering methods under the assumption that this cost is proportional to the number of corrections required, Figure 4 shows the cumulative count of the number of correction as training progresses. It shows the total number of corrections a user would have made after a given number of model retraining iterations.

The figure very clearly demonstrates the difference between the active and *a priori* methods, and the significantly higher number of total corrections required by active approaches. It is also interesting to note that there is little difference between random selection and (*a priori*) kernel herding. Although there are some cases in which the latter appears preferable—this seems to be the case when the more complex residual nets are applied to Flowers and Dogs—random selection is always competitive; in particular, it appears to outperform kernel herding on the sign language data. One can conjecture that this is because the sign language images are very different from the ImageNet data used to pre-train the convolutional neural networks providing the embeddings for *a priori* kernel herding.

So far, we have not examined directly the number of corrections that is needed to reach a certain level of accuracy, which is ultimately what matters in practical applications of the human-in-the-loop learning approaches we consider. Hence, in the last set of plots, in Figure 5, we plot classification accuracy, as estimated on the holdout data, with respect to the number of corrections that are required to achieve that accuracy, for each of the ordering strategies, models, and datasets. In these plots, the ideal method would be towards the top-left corner of the plot. The difference between the curves on the y-axis shows the difference in accuracy achieved for the same number of total corrections performed by the user. These plots indicate quite clearly that *a priori* kernel herding is the method of choice: for three of the datasets, and excluding the results for Mobilenet, it yields a noticeably higher accuracy for a given number of corrections than the other ordering methods. Simple random sampling comes second and performs as well as *a priori* kernel herding—or even slightly better—on the sign language data and when using Mobilenet. However, when it performs better, the difference is very small. The active learning methods perform worse throughout.

7 CONCLUSION

In this paper, we studied how to order batches of data in an iterative training scheme to achieve the highest model performance while minimising the effort needed by a human in the loop to correct misclassified examples. Experiments were conducted based on five methods of ordering, across four image classification datasets, and using three popular pre-trained models. Two of the methods ordered the examples *a priori* the other three employed an active learning approach.

We conclude that it is important to consider accuracy of the model in relation to the number of corrections that are required: using accuracy in relation to the number of labelled training examples—as is common practice in the literature—can be misleading. More specifically, active learning methods require more cumulative corrections than *a priori* methods for a given level of accuracy. Our results indicate that random sampling is a competitive strategy for some neural network architectures and datasets. However, *a priori* kernel herding based on the embeddings produced by convolutional neural networks pre-trained on ImageNet yields the best performance overall and can noticeably reduce the number of corrections required for a given level of accuracy when compared to random sampling.

There are a number of avenues for future work. In particular, it is important to build trust in human-in-the-loop scenarios, and deep networks are notorious for focusing on the wrong things when performing image classification [15]. One approach in the context of this paper is to study whether "single-object" object detection—which could provide a bounding box in addition to a class label—would be helpful because this could indicate to the domain expert whether the model is focusing on the correct area of the image. Human-in-the-loop systems offer the possibility of monitoring and correcting model bias, and this is certainly something that needs to be explored. Another avenue is to make use of advances in methods that explain deep model decision-making [18], making them part of the human-in-the-loop experience.

ACKNOWLEDGMENTS

The work contained here is part of a user-friendly deep learning project funded by the New Zealand Ministry for Business, Innovation and Employment.

REFERENCES

[1] Albert Bifet, Ricard Gavaldà, Geoff Holmes, and Bernhard Pfahringer. 2018. *Machine Learning for Data Streams with Practical Examples in MOA*. MIT Press. https://moa.cms.waikato.ac.nz/book/.

[2] Samuel Budd, Emma C Robinson, and Bernhard Kainz. 2021. A survey on active learning and human-in-the-loop deep learning for medical image analysis. *Medical Image Analysis* (2021), 102062.

[3] Yutian Chen, Max Welling, and Alex Smola. 2010. Super-samples from kernel herding. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*. 109–116.

[4] Michael Desmond, Zahra Ashktorab, Michelle Brachman, Kristina Brimijoin, Evelyn Duesterwald, Casey Dugan, Catherine Finegan-Dollak, Michael Muller, Narrendra Nath Joshi, Qian Pan, and Aabhas Sharma. 2021. Increasing the speed and Accuracy of Data Labelling Through an AI Assisted Interface. In *Proceedings of the Twenty-Sixth ACM Conference on Intelligent User Interfaces, IUI 2021*. 392–401.

[5] E. Frank, M. A. Hall, G. Holmes, R. Kirkby, B. Pfahringer, and I. H. Witten. 2005. *Weka: A machine learning workbench for data mining*. Springer, Berlin, 1305–1314. http://researchcommons.waikato.ac.nz/handle/10289/1497

[6] Antonio Gulli and Sujit Pal. 2017. *Deep learning with Keras*. Packt Publishing Ltd.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[8] Tharindu Kaluarachchi, Andrew Reis, and Suranga Nanayakkara. 2021. A Review of Recent Deep Learning Approaches in Human-Centered Machine Learning. *Sensors* 21, 7 (2021), 2514.

[9] Andreas Kirsch, Sebastian Farquhar, and Yarin Gal. 2021. A Simple Baseline for Batch Active Learning with Stochastic Acquisition Functions. *arXiv preprint arXiv:2106.12059* (2021).

[10] David D Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *International Conference on Machine learning*. Morgan Kaufmann, 148–156.

[11] Brendon Lutnick, Brandon Ginley, Darshana Govind, Sean D McGarry, Peter S LaViolette, Rabi Yacoub, Sanjay Jain, John E Tomaszewski, Kuang-Yu Jen, and Pinaki Sarder. 2019. An integrated iterative annotation technique for easing neural network training in medical image analysis. *Nature machine intelligence* 1, 2 (2019), 112–119.

[12] Koki Madono, Teppei Nakano, Tetsunori Kobayashi, and Tetsuji Ogawa. 2020. Efficient Human-In-The-Loop Object Detection using Bi-Directional Deep SORT and Annotation-Free Segment Identification. In *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 1226–1233.

[13] Mohammad Amin Morid, Alireza Borjali, and Guilherme Del Fiol. 2020. A scoping review of transfer learning research on medical image analysis using ImageNet. *Computers in biology and medicine* (2020), 104115.

[14] Gonzalo Ramos, Christopher Meek, Patrice Simard, Jina Suh, and Soroush Ghorashi. 2020. Interactive machine teaching: a human-centered approach to building machine-learned models. *Human–Computer Interaction* 35, 5-6 (2020), 413–451.

[15] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.

[16] Soumya Roy, Asim Unmesh, and Vinay P Namboodiri. 2018. Deep active learning for object detection.. In *BMVC*, Vol. 362. 91.

[17] Burr Settles. 2012. *Active Learning*. Morgan & Claypool Publishers.

[18] Amitojdeep Singh, Sourya Sengupta, and Vasudevan Lakshminarayanan. 2020. Explainable deep learning models in medical image analysis. *Journal of Imaging* 6, 6 (2020), 52.

[19] Bharath K Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Gert Lanckriet, and Bernhard Schölkopf. 2008. Injective Hilbert space embeddings of probability measures. In *21st Annual Conference on Learning Theory (COLT 2008)*. Omnipress, 111–122.

[20] Nima Tajbakhsh, Jae Y Shin, Suryakanth R Gurudu, R Todd Hurst, Christopher B Kendall, Michael B Gotway, and Jianming Liang. 2016. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging* 35, 5 (2016), 1299–1312.

[21] Jesper E Van Engelen and Holger H Hoos. 2020. A survey on semi-supervised learning. *Machine Learning* 109, 2 (2020), 373–440.

[22] Jeffrey S Vitter. 1985. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)* 11, 1 (1985), 37–57.

[23] Xingjiao Wu, Luwei Xiao, Yixuan Sun, Junhang Zhang, Tianlong Ma, and Liang He. 2021. A Survey of Human-in-the-loop for Machine Learning. *arXiv preprint arXiv:2108.00941* (2021).